

高等学校计算机教材建设立项项目

高等学校计算机专业规划教材

数字签名与安全协议

任　伟　编著

清华大学出版社

高等学校计算机教材立项项目
高等学校计算机专业规划教材

数字签名与安全协议

任 伟 编著

清华大学出版社
北 京

内 容 简 介

本书内容包括四个部分：基本数字签名(基于单向性的签名、基于离散对数的签名、基于离散对数签名的扩展讨论、基于身份识别协议的签名)，高级数字签名(盲签名、代理签名、多重数字签名、环签名、指定验证者签名等)，安全协议(实体认证协议、身份识别协议、密钥协商协议、比特承诺、零知识证明协议、不经意传输、秘密共享、安全多方计算)，基于身份的密码学和可证明安全性。本书的特点是注重介绍密码学方案的构造逻辑、设计规律，在给出方案的同时，还给出具有启发性的解释和讨论，解释方案的设计机理和思路，以培养学习者的逻辑推理能力。

本书主要面向高等学校信息安全、密码学、电子对抗、应用数学、计算机科学、通信工程、信息工程、软件工程等专业本科高年级学生和研究生，对具有密码学基础的研究人员也有启发作用和参考价值。

图书在版编目(CIP)数据

数字签名与安全协议/任伟编著. —北京：清华大学出版社，2015
高等学校计算机专业规划教材
ISBN 978-7-302-39746-5

Ⅰ. ①数…　Ⅱ. ①任…　Ⅲ. ①密码协议　Ⅳ. ①TN918.1

中国版本图书馆 CIP 数据核字(2015)第 072211 号

责任编辑：龙启铭　战晓雷
封面设计：
责任校对：梁　毅
责任印制：

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：10.5　　**字　　数**：259 千字
版　　次：2015 年 8 月第 1 版　　**印　　次**：2015 年 8 月第1 次印刷
印　　数：1～　000
定　　价：25.00 元

产品编号：057786-01



前言

“密码学”是网络空间安全、信息安全、密码学、电子对抗与网络攻防等专业中一门重要的专业课，在学科知识体系中占据重要的地位。“数字签名与安全协议”把密码学基础知识融会贯通，是密码学中重要的应用部分，内容十分丰富。

作者在密码学课程教学过程中发现，很多学生只是停留在对密码学方案过程的记忆上，局限于记忆密码学方案的具体步骤，而未能思考和总结密码学方案的设计方法和一般规律。目前相关书籍在密码学设计机理等内在机制的讲解方面仍然有改进的空间，在思维的启发性以及学生创造性能力培养方面仍然有待完善。本书试图在这些方面做一点抛砖引玉的尝试，写作时遵循了以下思路：

(1) 注重启发性。注重对设计原理的分析以及对设计的动机和逻辑性的解释，让学习者知其然，也知其所以然。

(2) 注重知识点的逻辑联系和类比。注重章节间和章节内前后各个分离的知识点间的联系和类比关系，明确给出各知识点间的关联并加以强调，便于读者体会密码算法或协议设计的奥妙。大量采用类比法、比较法、归纳法、图示法，试图使读者对所学内容能够反复巩固，前后联系。

(3) 注重原理的总结和推广。在介绍具体构造方案后，给出一般性构造方案，或者加以讨论和提炼总结，有利于知识的理解和举一反三。

(4) 广度和深度兼具。基本原理、基本概念的讲解力求透彻，有深度。通过扩展阅读加大广度，便于回顾经典论文或者了解国内外发展动态。

全书共15章，分为4个部分，第1部分“基本数字签名”包括第1章“数字签名概述”、第2章“基于单向性的签名”、第3章“基于离散对数的签名”、第4章“离散对数签名的扩展*”、第5章“基于身份识别协议的签名*”。第2部分“高级数字签名”包括第6章“盲签名”、第7章“代理签名”、第8章“多重数字签名”、第9章“其他高级签名*”。第3部分“安全协议”包括第10章“实体认证协议”、第11章“身份识别协议”、第12章“密钥协商协议”、第13章“高级协议”。第4部分“基于身份的密码学和可证明安全性”包括第14章“基于身份的公钥密码学”和第15章“可证明安全签名和协议*”。

全书精心安排了示例。为帮助读者进一步对内容的拓展研究，有针对性地提供了扩展阅读建议，用于开展课外学习和论文研读讨论。每部分的小结归纳了本部分中的知识点，并指出重点和难点，便于复习。打*号的章

节可选学。

本书主要面向高等学校网络空间安全、信息安全、密码学、电子对抗、应用数学、计算机科学、通信工程、信息工程、软件工程等专业本科高年级学生和研究生，对具有密码学基础的研究人员也有启发作用和参考价值。

本书获得全国高等学校计算机教育研究会2013年度高等学校计算机教材建设立项资助，湖北省教育厅高等学校教学研究重点项目，以及国家自然科学基金面上项目(61170217)的资助，在此表示感谢。感谢研究生孙亚璐、林佳华、曹强的辅助工作。由于作者水平有限，在此衷心希望读者不吝赐教。

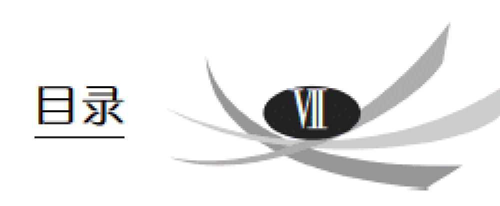
作　者
2015年4月



第1部分　基本数字签名

第2部分 高级数字签名

第 3 部分 安全协议


Ⅶ

第 1 部分

基本数字签名

第 1 章 数字签名概述

1.1 数字签名的一般模型

随着计算机网络的发展，特别是电子商务的兴起，需要对消息进行消息完整性保护，对消息源进行鉴别，对交易进行认证，以及提供不可抵赖性保障。数字签名是手写签名的数字化形式。手写签名的基本特点是：能与被签名的信息在物理上不可分割，签名者不能否认自己的签名，签名不能伪造，签名容易被验证。数字签名是一串二进制数，应与被签名的信息"绑定"在一起。通常，数字签名应具有以下特性：

(1) 签名是可信的。任何人都可以验证签名的有效性。

(2) 签名是不可伪造的。除合法的签名者之外，任何其他人伪造签名是困难的。

(3) 签名是不可复制的。对某个消息的签名不能通过复制变为对另一个消息的签名。如果对某个消息的签名是从别处复制得到的，则任何人都可以发现签名和消息不一致，从而可以拒绝签名的消息。

(4) 签名的消息是不可改变的。经签名的消息不能被篡改。一旦已签名的消息被篡改，则任何人都可以发现消息和签名之间的不一致性。

(5) 签名是不可抵赖的。签名者事后不能否认自己的签名。

定义 1.1 一个数字签名方案是一个 5 元组 $(M,S,K,\mathrm{SIGN},\mathrm{VRFY})$，满足如下的条件：

(1) M 是一个可能消息的有限集。

(2) S 是一个可能签名的有限集。

(3) 密钥空间 K 是一个可能密钥的有限集。

(4) 对每一个 $k=(k_s,k_v)\in K$，都对应一个签名函数 $\mathrm{Sign}_{k_s}\in\mathrm{SIGN}$ 和验证算法 $\mathrm{Vrfy}_{k_v}\in\mathrm{VRFY}$。每一个 $\mathrm{Sign}_{k_s}:M\to S$ 和验证函数 $\mathrm{Vrfy}_{k_v}:M\times S\to\{\mathrm{True},\mathrm{False}\}$ 是一个对任意消息 $m\in M$ 和任意签名 $s\in S$ 满足下列方程的函数：

$$\mathrm{Vrfy}(m,s)=\begin{cases}\mathrm{True} & s=\mathrm{Sign}_{k_s}(m)\\ \mathrm{False} & s\neq\mathrm{Sign}_{k_s}(m)\end{cases}$$

对每一个 $k\in K$，函数 Sign_{k_s} 和 Vrfy_{k_v} 都是多项式时间可计算的函数。Vrfy_{k_v} 是一个公开函数，k_v 为公钥（验证密钥）；Sign_{k_s} 是一个密码函数，k_s 为私钥（签名密钥），要秘密保存。

1.2 数字签名的分类

基于数学难题的分类有基于离散对数问题的签名方案、基于大整数素数因子分解的签名方案、基于椭圆曲线离散对数问题的签名方案、基于二次剩余问题的签名方案。

基于数字签名是否具有恢复特性的分类有不具有消息恢复(message recovery)特性的签名和具有消息恢复特性的签名。

基于不同的加密方法的分类有基于对称加密算法的数字签名和基于公钥加密算法的数字签名。

基于签名用户的分类有单用户签名和多用户签名。多个用户的签名方案又称为多重签名方案。根据签名的过程不同,多重数字签名方案可分为有序多重数字签名方案和广播多重数字签名方案。

基于签名人对消息是否可见的分类:通常签名都是签名人对消息可见的,而盲签名方案表示签名者对消息不可见,但事后可以证明消息的存在。盲签名又根据签名者是否可以对消息者进行追踪分为弱盲签名方案和强盲签名方案。

基于签名人是否受别人委托签名的分类有普通签名方案和代理签名方案。如果授权的不是一个人,而是多个人,这时称为代理多重数字签名方案。

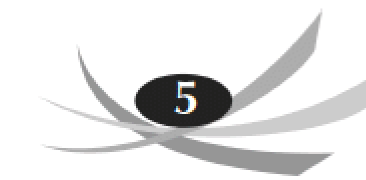
基于签名是否有仲裁的分类有直接数字签名和仲裁数字签名。直接数字签名是在签名者和签名接收者之间进行的,假设签名接收者知道签名者的公钥。仲裁数字签名在签名者、签名接收者和仲裁者之间进行。签名者和签名接收者共同信任仲裁者。签名者首先对消息进行数字签名,然后送给仲裁者。仲裁者首先对签名者送来的消息和签名进行验证,并对验证过的消息和签名附加一个验证日期和一个仲裁说明,然后把验证过的签名和消息发送给签名接收者。因为有仲裁者的验证,所以签名者无法否认自己签过的数字签名。

1.3 数字签名的设计原理*

数字签名的设计主要依靠 3 种方法:单向陷门函数、从身份识别协议通过非交互零知识证明的机制转化而来的知识签名、利用可交换的公钥加密直接构造。

在利用单向陷门函数的数字签名中,陷门信息作为签名人的私钥,签名人对私钥的拥有表明签名的真实性。这种基于单向陷门函数的数字签名依据的是两条基本的假设:一是私钥是安全的,只有其拥有者才能获得;二是产生数字签名的唯一途径是使用私钥。尽管数字签名的安全性并没有得到证明,但超出这种假设,即使用未知的密钥而非私钥,或使用未知的算法而非数字签名算法得到的结果被公钥验证成功的例子尚未出现。因此,这两条假设的破坏是计算上不可行的,因而这两条假设被认为是成立的。数字签名的假设和例外如图 1.1 所示。

第二种是从身份识别协议通过非交互零知识证明的机制转化而来的知识签名。它从身份识别协议转化而来,转化的方法主要是利用非交互零知识的方法(也称为 Fiat-



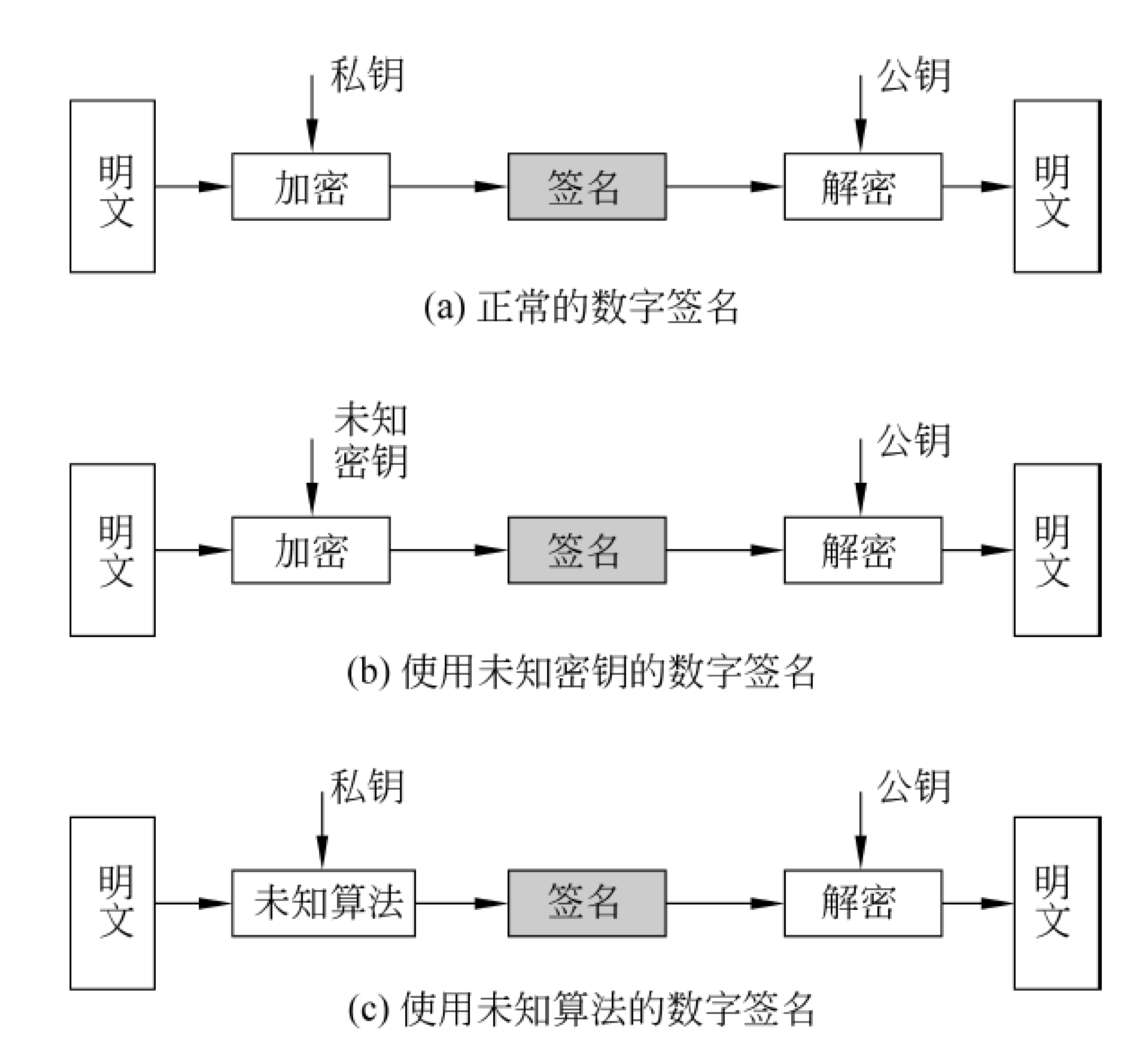


图 1.1　数字签名的假设和例外

Shamir 启发式)。具体而言,通常包含 3 个部分,利用了单向函数的单向性构造一个承诺机制,对一个随机数进行承诺(承诺后不能再改变),然后利用哈希函数的单向性构造随机挑战值,该挑战值通常是消息和承诺值两者连接后的散列值,签名者利用对秘密知识的拥有构造相应的应答。该应答即是签名,签名接收者接收签名后验证签名的有效性。

还有一类签名可以利用可交换的公钥加密直接构造。可交换的公钥加密系统是指:设 E_e 是一个公钥加密算法,有消息空间 M 和密文空间 C,设 $M=C$。令 D_d 是对应 E_e 的解密算法,因 E_e 和 D_d 都是置换,且有

$$D_d(E_e(m)) = E_e(D_d(m)) = m, m \in M$$

称这种类型的公钥加密方案是可交换的。

于是可以简单地通过解密算法进行签名,如 RSA 签名方案和 Rabin 签名方案。

1.4　数字签名的安全性*

1988 年,Shafi Goldwasser、Silvio Micali 和 Ronald Rivest 第一次严格定义了数字签名的安全性,并提出了 GMR 签名方案,是第一个可证明满足选择消息攻击下的存在性不可伪造的签名。

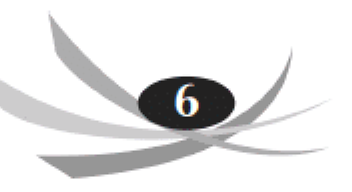
本节只作简要介绍,以便读者理解后文中的一些概念。

首先需要明确安全需求,才能设计安全的数字签名方案。在明确安全需求之前,需要先明确敌手模型。

敌手的能力(假想攻击方为 Oscar,通信方为 Alice):

(1) 唯密钥攻击(key-only attack)。敌手拥有公钥以及签名验证函数 $\mathrm{Vrfy}_{pk}()$。

(2) 已知消息攻击(known message attack)。敌手 Oscar 拥有 Alice 已签署的一系列消息签名的列表,例如 (x_1, y_1),(x_2, y_2),…,其中 x_i 是消息,y_i 是 Alice 对这些消息的签



名(有 $y_i=\mathrm{Sign}_{sk}(x_i)$,$i=1,2,\cdots$)。

(3) 选择消息攻击(chosen message attack)。Oscar 请求 Alice 对一系列消息的签名,她选择消息 $x_1,x_2,\cdots$,Alice 提供对这些消息的签名,它们分别为 $y_i=\mathrm{Sign}_{sk}(x_i)$,$i=1,2,\cdots$。选择只能在挑战消息发出之前进行。

有的文献将选择消息攻击进一步划分为一般选择消息攻击和定向选择消息攻击。两者的区别是:前者选择消息是在看到 Alice 的公钥之前,独立于 Alice 的公钥;后者是在看到 Alice 的公钥之后选择消息。

(4) 自适应选择消息攻击(adaptive chosen message attack)。敌手可以选择 x_i,而且允许根据先前的访问结果进行后续的选择(选择可以在挑战消息发出之后继续)。

在后面,不再区分攻击(3)和(4),将其统称为选择消息攻击。

敌手的目标:

(1) 完全攻破(total break)。敌手能够确定 Alice 的私钥,即签名函数 $\mathrm{Sign}_{sk}()$,从而对任何消息都产生有效的签名。

(2) 选择性伪造(selective forgery)。敌手以某个不可忽略的概率对另外某个人选择的消息产生一个有效的签名。也就是说,如果给敌手一个消息 x,她能(以某种概率,该概率不可忽略)确定签名 y,使得 $\mathrm{Vrfy}_{pk}(x,y)=\mathrm{true}$,且该消息不是 Alice 曾经签名过的消息。

(3) 存在性伪造(existential forgery)。敌手至少能够为一则消息产生一个有效的签名。换句话说,敌手能产生一个消息和签名对(x,y),其中 x 是消息,y 是签名,而 $\mathrm{Vrfy}_{pk}(x,y)=\mathrm{true}$。该消息不是 Alice 曾经签名过的消息。

一个签名方案不可能是无条件安全的,因为对一个给定的消息 x,Oscar 使用公开算法 Vrfy_{pk} 测试所有可能的签名 y,直到发现一个有效的签名。因此,给定足够的时间,Oscar 总能对任何消息伪造 Alice 的签名。因此,和公钥密码体制一样,目标是找到计算上安全的签名方案。

第 2 章

基于单向性的签名

2.1 基于单向函数的签名

首先介绍一次性签名方案，因为其非常简洁，且具有构造签名的朴素思想。

2.1.1 Lamport 一次签名

1979 年，L. Lamport 提出基于任意单向函数的一次签名方案。该方案在单向函数是双射的条件下是可证明唯密钥攻击安全的。

所谓一次签名是指一对公、私钥只能用于对一个消息进行签名。每次签署消息都需要更新公钥。由于这种签名往往依赖单向函数或对称加密密钥算法，因而具有签名生成和签名验证高效的特点，故可应用在芯片卡等计算能力较低的环境。

Lamport 一次签名方案如下：

（1）密钥生成。设 k 是一个正整数。假定 $f: Y \to \mathbb{Z}$ 是一个单向函数，设随机选择的 $y_{i,j} \in Y, 1 \leqslant i \leqslant k, j=0,1$。设 $z_{i,j}=f(y_{i,j}), 1 \leqslant i \leqslant k, j=0,1$。密钥 K 由 $2k$ 个 y 和 $2k$ 个 z 构成，y 是私钥，z 是公钥。

（2）签名生成。对于 $K=(y_{i,j}, z_{i,j}: 1 \leqslant i \leqslant k, j=0,1)$，定义

$$s = \mathrm{Sign}_K(x_1, x_2, \cdots, x_k) = (y_{1,x_1}, y_{2,x_2}, \cdots, y_{k,x_k})$$

（3）签名验证。关于消息 $(x_1, x_2, \cdots, x_k)$ 的签名 $(s_1, s_2, \cdots, s_k)$ 验证如下：

$$\mathrm{Vrfy}_K((x_1, x_2, \cdots, x_k), (s_1, s_2, \cdots, s_k)) = \mathrm{True} \Leftrightarrow f(s_i) = z_{i,x_i}$$

该签名方案的示意图如图 2.1 所示。根据消息的比特值，给出相应的私钥。非正式地说，有 $2k$ 套"锁-钥匙"对，即 $2k$ 把"锁"和 $2k$ 把"钥匙"。"锁"为公钥，"钥匙"为私钥（是秘密的）。明文中的每一位对应两套"锁-钥匙"对。根据明文中该位的比特为 0 或者为 1，亮出两把秘密"钥匙"中的一个。可见，一旦这把"钥匙"公开了，如果不更换"锁"，敌手就可以利用该"钥匙"伪造签名了。

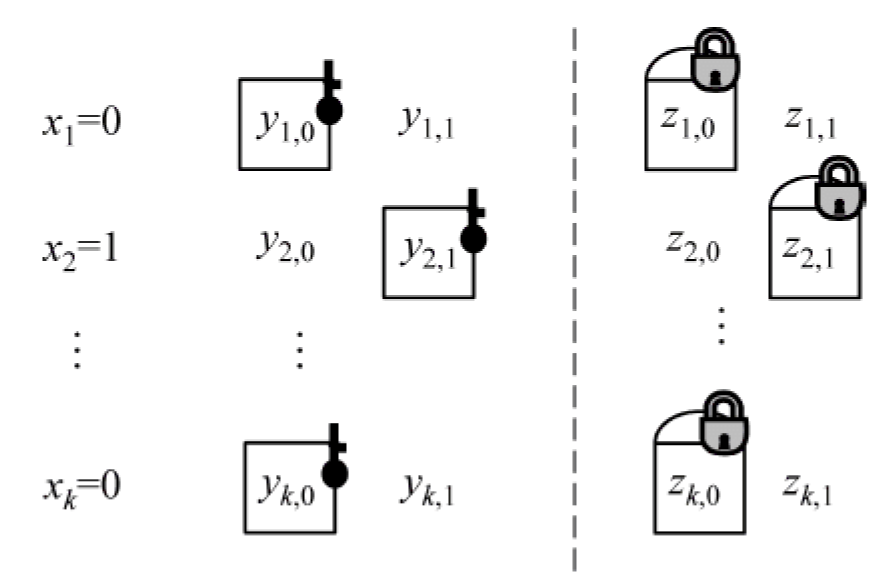


图 2.1　Lamport 一次签名方案

例 2.1　不妨设单向函数是 $f(x)=3^x \bmod 7879$，（3 是素数群 $\mathbb{Z}_{7879}^*$ 的生成元），假定明文消息有 3 个比特，即 $k=3$。取私钥为

$$y_{1,0} = 5831, \quad y_{1,1} = 735$$
$$y_{2,0} = 803, \quad y_{2,1} = 2467$$
$$y_{3,0} = 4285, \quad y_{3,1} = 6449$$

计算公钥为

$$z_{1,0}=3^{5831} \bmod 7879=2009,\quad y_{1,1}=3810$$
$$z_{2,0}=4672,\quad z_{2,1}=4721$$
$$z_{3,0}=268,\quad z_{3,1}=5731$$

假设需要签名的明文为(010)，那么签名为$(y_{3,0},y_{2,1},y_{1,0})=(4285,2467,5831)$。接收方验证签名：

$$3^{5831} \bmod 7879=2009$$
$$3^{2467} \bmod 7879=4721$$
$$3^{4285} \bmod 7879=268$$

从该例可见，如果在签名一次后不更换公钥，敌手若知道了 010 和 101 的签名，便可以伪造任何 3 比特消息的签名了，因为敌手拥有了所有的私钥。

可证明安全性。如果 f 是一个双射单向函数，那么 Lamport 一次签名方案满足唯密钥攻击条件下存在性不可伪造(EU-KOA)。

证明的思路：反证法。

如果 Lamport 一次签名方案不是唯密钥攻击条件下存在性不可伪造的，则 f 不是单向函数。这与原假设矛盾，故得证。下面给出具体证明。

如果 Lamport 一次签名方案不是唯密钥攻击条件下存在性不可伪造的，则存在 Lamport-Forge 算法，给定任意特定的公钥，输出一个伪造的签名。现在利用该算法作为子过程构造一个算法 F-Preimage，该算法对于任意选择的元素 $z\in\mathbb{Z}$，输出 z 关于 f 的原像。可见，这是一个使用 Lamport-Forge 算法作为预言机(Oracle)的归约。

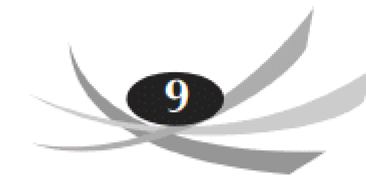
算法 2.1 F-Preimage(z)

```
/* 求 z 关于函数 f 的原像 */
/* 输入像 z */
/* 输出 z 的原像 */
{
    z_{1,0} = z, z_{1,1} <-r- Z
    z̄ = (z_{1,0}, z_{1,1})
    (x_1, a_1) <- Lamport-Forge(z̄)
    If x_1 = 0
        Return(a_1)
    Else
        Return("Failure")
}
```

这里$\xleftarrow{r}$表示随机选择，构造一个公钥 $\bar{z}=(z_{1,0},z_{1,1})$，作为预言机的问询，预言机返回一个伪造的签名，如果该签名恰好为公钥 $z_{1,0}=z$ 的签名(私钥)，则得到了 z 的原像。显然，这种情况的概率是 1/2。

2.1.2 基于对称加密的一次签名方案

一个有趣的现象是：最早的签名方案是根据对称加密算法构造的。



Rabin 的一次性签名方案是最早的签名方案之一，但该方案中签名的验证需要签名者和验证者合作才能完成。下面简要介绍 Rabin 的基于对称加密算法(如 DES)的一次性签名方案。

1. 密钥产生

产生签名的密钥。签名方随机产生 $2t$ 个密钥作为签名用的密钥，所以这些密钥需要保密，$2t$ 个密钥为 $K_1,K_2,\cdots,K_{2t}$。

产生用于验证签名的数据信息。首先产生 $2t$ 个随机数：$u_1,u_2,\cdots,u_{2t}$；然后用 K_i 分别加密随机数 $u_i,i=1,2,\cdots,2t$，得到 $2t$ 个密文数据：$U_1,U_2,\cdots,U_{2t}$，其中 $U_i=E(u_i,K_i),i=1,2,\cdots,2t$。公开这 $2t$ 个随机数和 $2t$ 个密文数据，作为用于验证签名的数据。需要注意的是，公布的 u_i 序列和 U_i 序列的排列顺序应与 $K_1,K_2,\cdots,K_{2t}$的顺序一致。

2. 签名过程

用 K_i 对报文 M 的压缩码 $h(M)$加密获得签名：$S=(E(h(M),K_1),E(h(M),K_2),\cdots,E(h(M),K_{2t}))$，简记为 $S=(S_1,S_2,\cdots,S_{2t})$，其中，$h$ 表示压缩函数，E 表示加密。签名者把消息和签名(M,S)发送给验证者。

验证方索取 t 个密钥。验证者收到签名消息后，随机产生一个长为 $2t$ 的比特串，要求其中包含 t 个比特 0，t 个比特 1。然后把该比特串发送给签名者。签名者收到该长为 $2t$ 的比特串之后，发送回 t 个密钥。发送原则是：如果比特串的第 i 位为 1，则签名方把 K_i 发送给对方。发送也要求按顺序完成。

3. 验证签名

验证方收到签名方的 t 个密钥之后，就可以验证签名的有效性。验证方法可以如下表示：

$$\begin{cases}E(u_i,K_i)=U_i\\E(h(M),K_i)=S_i\end{cases}\quad i=j_1,j_2,\cdots,j_t$$

其中，K_i 表示接收方收到签名方的 t 个密钥，$i=j_1,j_2,\cdots,j_t$。如果对于所有的 t 个密钥，上面两式均成立，则签名有效，否则签名无效。

作为最早的签名方案，其设计思路是证明签名者拥有对称加密方案中的密钥。一次性数字签名的缺点是要求对每一个消息都使用一个新的公钥。它的优点是签名产生和验证速度较快，特别适用于计算能力有限的芯片卡中。通常把它们与可信第三方相结合，并通过验证树结构来实现。

2.2 利用公钥加密的签名

221 Rabin 数字签名

Rabin 数字签名非常简单，方案描述如下：

(1) 密钥生成。两个随机选取的不同的模 4 余 3 的大素数 p 和 q，$n=pq$，其中 n 为公钥，p 和 q 需要保密。消息空间和签名空间都是由同为模 p 平方剩余和模 q 平方剩余的正整数构成的集合。

(2) 签名生成。消息 $m\in Z_n$ 签名时,首先要确保 m 既是 p 的平方剩余,又是 q 的平方剩余。如果 m 不能满足这一条件,可先对 m 做一个变换,将其映射成符合要求的 m'。假设对 m 签名,$s=\text{Sign}(m)=\sqrt{m} \bmod n$。

(3) 签名验证。$\text{Vrfy}(m,s)=\text{true}\Leftrightarrow m=s^2 \bmod n$。

Rabin 数字签名的安全性可证明等价于大整数分解的困难性。

Rabin 数字签名容易遭受选择消息攻击。攻击者选择一个 x,计算出 $x^2=m \bmod n$,将 m 发送给签名者签名,等待签名者返回的签名 s。易知,签名者对 m 的签名可能有 4 种结果,其中包括 $\pm x \bmod n$。因此,攻击者有 1/2 的概率得到一个非 $\pm x \bmod n$ 的签名,从而解出 p 和 q,达到分解 n 的目的,从而完全攻破签名体制。

另一个缺点是它要求被签名消息必须是模 n 的平方剩余,否则需要映射后签名,这给该方案的应用造成了很大的不便。后来,Rabin 又对方案进行了改进,使任何消息都能直接被签名,这里不再介绍。

222 RSA数字签名

RSA 签名方案是目前使用较多的一个签名方案,也是已经提出的数字签名方案中最容易理解和实现的签名方案,其安全性基于大整数因子分解的困难性。方案描述如下:

(1) 密钥生成。首先选取两个不同的大素数 p 和 q,计算 $n=pq$,其欧拉函数 $\phi(n)=(p-1)(q-1)$,然后随机选择整数 $e(1<e<\phi(n))$,满足 $\gcd(e,\phi(n))=1$,计算 $d=e^{-1} \bmod n$,则签名者 A 的公钥为(n,e),私钥为 d。p 和 q 是秘密参数,需要保密,如不需要保存,计算出 e、d 后销毁 p、q。

(2) 签名生成。待签名的消息为 $m\in Z_n$,签名为 $s=\text{Sign}(m)=m^d \bmod n$。将$(m,s)$发送给接收者。

(3) 签名验证。签名接收者 B 收到消息 m 和签名 s 后,验证等式 $m=s^e \bmod n$ 是否成立。若成立,签名有效;否则签名无效。

正确性证明

$$\begin{aligned} s^e &= \bmod n = m^{ed} \bmod n = m^{h\phi(n)+1} \bmod n = m(m^{\phi(n)})^h \bmod n \\ &= m\times 1^h \bmod n = m \end{aligned}$$

1. 安全性分析

上述方案是一个最简单的 RSA 签名方案(教课书 RSA 签名方案),但是不难发现,上述方案存在一些问题:

(1) 任何人都可以随机选择一个任意的 $y\in Z_n$,计算 $x=y^e \bmod n$,于是任何人都可以伪造一个有效的消息签名对(x,y)。这种情况就是唯密钥攻击条件下的存在性伪造。

(2) 如果消息 x_1,x_2 的签名分别为 y_1,y_2,则任何知道 x_1,y_1,x_2,y_2 的人都可以伪造对消息 x_1x_2 的签名 y_1y_2,即生成一对有效的明文和签名对(x_1x_2,y_1y_2)。因为$(y_1y_2)^e=(y_1^e)(y_2^e)=x_1x_2$,签名验证成立。这种情况就是选择消息攻击下的选择性伪造,或者是已知消息攻击条件下的存在性伪造。

(3) 这种签名方案对签名的消息有限制,即 $x\in Z_n$,于是长度不能超过$\lfloor \log_2 n \rfloor$比特。通常实际应用中要签名的消息都较长,可能比 n 大。这种情况下,只能先对消息进行分组,然

后对每组消息分别进行签名，这样导致签名长度变长，签名速度变慢，签名验证更耗时。

(4) 利用签名服务获取明文(消息破译)。假设敌手已经截获密文 c，$c=x^e \bmod n$，想求出明文 x。于是，先选择一个小的随机数 r，计算

$$s = r^e \bmod n$$
$$l = s \times c \bmod n = x^e r^e \bmod n$$
$$t = r^{-1} \bmod n$$

因为 $s=r^e$，所以 $s^d \bmod n=(r^e)^d \bmod n=r \bmod n$。然后，敌手设法让签名者对 l 签名，于是敌手又获得 $k=l^d \bmod n$。敌手计算 $t\times k=r^{-1}\times l^d \equiv r^{-1}\times s^d \times c^d \equiv r^{-1}\times r\times c^d \equiv c^d \equiv x \bmod n$，于是敌手可获得明文 x。

(5) 先加密后签名方案的攻击。这种情况的原理与情况(2)类似，是选择消息攻击下的选择性伪造，或者是唯密钥攻击条件下的存在性伪造。

假设签名者 A 采用先加密后签名的方案发送消息 x 给接收者 B，即先用 B 的公开密钥 e_B 对 x 加密，然后用自己的私钥 d_A 签名。设 A 的模数为 n_A，B 的模数为 n_B，于是 A 发送给 B 的加密后签名为$(x^{e_B} \bmod n_B)^{d_A} \bmod n_A$。

B 可以伪造 A 的签名，方式是改变加密的消息。由于 B 知道 n_B 的分解，于是能够计算模 n_B 的离散对数，即能够找到 k 满足$(x_1)^k=x \bmod n_B$。

然后，公开新公钥为 ke_B，声称收到的消息为 x_1，而不是 x。

由于$(x_1^{ke_B} \bmod n_B)^{d_A} \bmod n_A=(x^{e_B} \bmod n_B)^{d_A} \bmod n_A$，A 无法否认。

上述问题中的(2)、(4)和(5)均来自签名函数的同态特性，即两个签名的乘积等于另一个消息的签名，也就是说 $s_1 s_2=m_1^d m_2^d=(m_1 m_2)^d \bmod n$。为了去掉这一特性，常见的方法是在签名之前先对消息做散列变换，然后对变换后的消息进行签名。即 $s=\mathrm{Sign}(m)=h(m)^d \bmod n$，这里 $h(m)$是一个散列函数。签名验证也做相应的改动，即先验证等式 $h(m)=s^e \bmod n$ 是否成立。

思考 2.1　为什么经过散列变换后，攻击(1)和(2)不会发生？

攻击(1)中，即使敌手任选 y 计算出 $t=y^e \bmod n$，由于散列函数的单向性，无法找到 x 使得 $h(x)=t$。

攻击(2)中，即使敌手拥有消息 x_1，x_2 的签名 y_1，y_2，要想使得构造的签名 $y_1 y_2$ 为某个消息 t 的签名，需要寻找 t，使其满足 $h(t)=(y_1 y_2)^e \bmod n$，由于散列函数的单向性，这是困难的。

同理，可说明(4)和(5)不再会发生。

因此，一个典型的签名范例如图 2.2 所示。

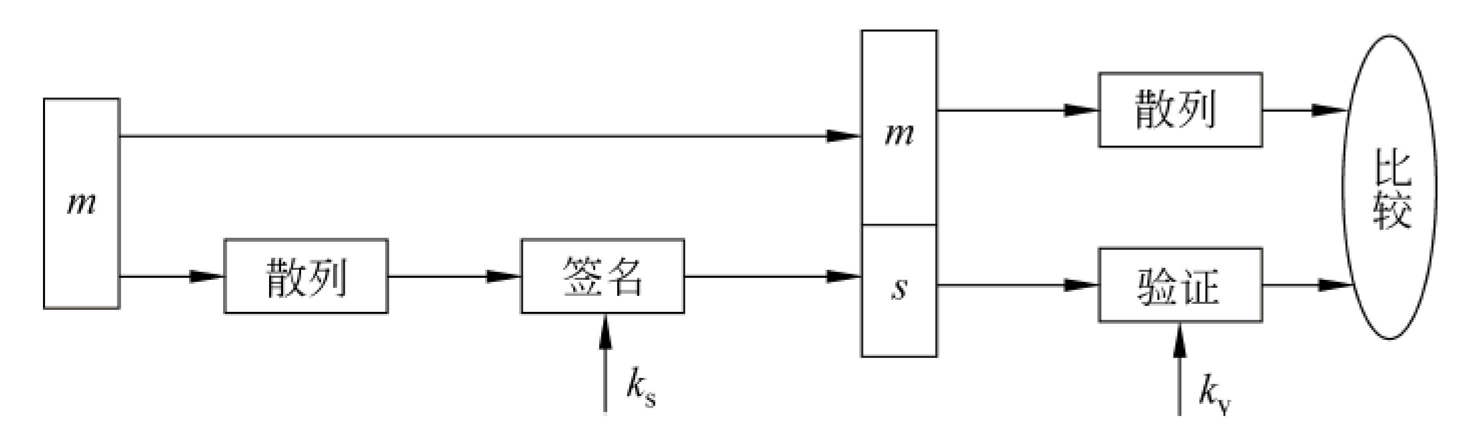


图 2.2　先散列后签名

2．散列后签名方案的一般安全性分析

类似上面的方法，将待签名的消息先计算散列值，然后进行签名的方法，可一般性地描述为 $s=\mathrm{Sign}(h(m))$。下面讨论该类方法的安全性：

1）已知消息攻击的存在性伪造

假设敌手有一个有效的数字签名 (m,s)，其中 $s=\mathrm{Sign}_k(h(m))$。敌手首先计算 $h(m)$，如果能够找到 $m'\neq m$，使得 $h(m')=h(m)$，则 (m,s) 就是一个有效的签名，完成了一次存在性伪造。为了阻止该攻击，必须要求散列函数 h 是抗第二原像的(抗弱碰撞的)。

2）选择消息攻击的存在性伪造

敌手首先找到 $m'\neq m$，使得 $h(m)=h(m)$。然后将消息 m 发送给签名者，并让 A 对消息的散列值 $h(m)$ 签名，从而得到 $s=\mathrm{Sign}_k(h(m))$，于是敌手可以成功伪造签名 (m,s)，完成了选择消息攻击条件下的存在性伪造。为阻止该类攻击，必须要求散列函数 h 是抗强碰撞的。

3．RSA 数字签名与 RSA 加密结合使用时的问题

实际应用中，常常需要将签名和加密结合使用，由于不同的用户使用的模数步一样，使用中需要注意结合的顺序。设 RSA 密码系统中，用户 A 的公钥为 (N_A,e_A)，私钥为 d_A，用户 B 的公钥为 (N_B,e_B)，私钥为 d_B，且设 $N_A<N_B$，则用户 A 若想发送一个既签名又加密的消息 m 给 B，应采取的步骤如下：

(1) 利用自己的签名密钥 d_A 对消息 m 进行签名，得到的签名为 $y=m^{d_A} \bmod N_A$。

(2) 利用接收方 B 的公钥对其加密得到 $c=y^{e_B} \bmod N_B$，得到的密文 c 中包含了对消息 m 的签名。

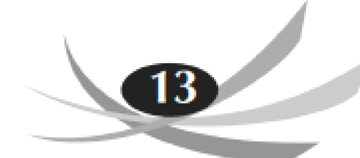
(3) 接收方 B 在收到密文后，先解密，然后验证签名。

上述过程在 $N_A>N_B$ 时，会导致解密后的结果不能通过签名验证。例如，对于参数 $(N_A=13\times17=221, e_A=29)$，$(N_B=11\times7=77, e_B=7)$，消息 $m=100$，m 的签名为 $y=100^5 \bmod 221=172$，用 B 的公钥加密后得 $c=172^7 \bmod 77=39$，B 收到密文 39 后，解密得到 $y'=18 \bmod 77$，验证 $m'=18^{29} \bmod 221=81$，于是，$m'\neq m$。易知，发生这种情况的概率为 $(N_A-N_B)/N_A$。

为解决上述问题，常用的一个方法是先加密后签名，即保证先进行模数较小的运算。但该方法容易遭受攻击，攻击者可对加密的消息进行签名，扰乱了消息来源。另一个办法是系统中每个用户采用两个模数，一个用于加密，另一个用于签名，且保证所有用户的签名模数均小于其他用户的加密模数。这样，确保了先签名后加密不会出现问题。如可选择所有签名的模数为 t 比特，所有加密的模数为 $t+1$ 比特。

思考 2.2 作为一个有趣的扩展思考，是否存在多个密钥的公钥密码体制？

可以将 RSA 体制推广为有多个密钥的公钥体制。令 $n=pq$，p 和 q 为素数，今选择 t 个密钥 $k_1,k_2,\cdots,k_t$ 使 $k_1k_2\cdots k_t\equiv1 \bmod (p-1)(q-1)$。令 M 是明文，于是有 $M^{k_1k_2\cdots k_t}\equiv M \bmod n$，因此可以由 $k_1,k_2,\cdots,k_t$ 组成多种加解密组合。例如，$t=5$，则可以用 k_2,k_4 加密，有密文 $C=M^{k_2k_4} \bmod n$；用 k_1,k_3,k_5 解密，有 $M=C^{k_1k_3k_5} \bmod n$。类似地，可以扩展到



多签名体制。例如，选择 $t=3$，可将 k_1 作为 A 的签名私钥，k_2 是 B 的签名私钥，k_3 是验证公钥。对一个给定文件，可由 A 进行签名，得 $S'=M^{k_1} \bmod n$，然后将 S' 传送给 B 签名，B 先恢复消息以证实文件内容：$M=S'^{k_2k_3} \bmod n$，然后对其作进一步签名：$S=S'^{k_2} \bmod n$，S 就是 A 和 B 的签名结果，任何人都可用公钥 k_3 验证签名：$M=S^{k_3} \bmod n$。

第 3 章 基于离散对数的签名

3.1 ElGamal 签名

3.1.1 ElGamal 签名体制

1985 年，T. ElGamal 提出一个基于离散对数问题的数字签名体制，称为 ElGamal 数字签名体制，其安全性基于有限域上的离散对数问题的困难性。方案描述如下：

(1) 密钥生成：选取大素数 p，$g \in \mathbb{Z}_p^*$ 是一个本原元。p，g 为公开参数。随机选取整数 x，$1 \leqslant x \leqslant p-2$，计算 $y = g^x \bmod p$。y 是公钥，x 是私钥。

(2) 签名生成：设 $m \in \mathbb{Z}_p^*$ 是待签名的消息。随机选取一个秘密整数 k，$1 \leqslant k \leqslant p-2$，对消息 m 的签名为 $\mathrm{Sign}(m) = (r, s) \in \mathbb{Z}_p^* \times \mathbb{Z}_{p-1}$，其中

$$r = g^k \bmod p$$
$$s = (m - xr)k^{-1} \bmod (p-1)$$

(3) 签名验证：对于 $m \in \mathbb{Z}_p^*$，$(r, s) \in \mathbb{Z}_p^* \times \mathbb{Z}_{p-1}$，如果

$$y^r r^s \equiv g^m \bmod p$$

则确认 (r, s) 为消息 m 的有效签名。

正确性证明

由于 $s = (m - xr)k^{-1} \bmod (p-1)$，于是 $xr + ks = m \bmod (p-1)$。因此，如果 $(r, s) \in \mathbb{Z}_p^* \times \mathbb{Z}_{p-1}$ 是消息的正确签名，则一定有 $y^r r^s \equiv g^{xr} g^{ks} \equiv g^m \bmod p$。

例 3.1 设 $p = 11$，$g = 2$ 是 $\mathbb{Z}_{11}$ 的本原元，选取 $x = 8$，计算 $y = g^x \bmod p = 2^8 \bmod 11 = 3$。$p$，$g$，$y$ 公开，x 保密。

假设 Alice 要对消息 $m = 5$ 签名，Alice 首先秘密选择 $k = 9$，因为 $\gcd(9, 10) = 1$，所以 9 模 10 的逆一定存在，有 $9^{-1} \bmod 10 = 9$。Alice 计算

$$r = g^k \bmod p = 2^9 \bmod 11 = 6$$
$$s = (m - xr)k^{-1} \bmod (p-1) = (5 - 8 \times 6) \times 9 \bmod 10 = 3$$

$(r, s) = (6, 3)$ 为 Alice 对消息 $m = 5$ 的签名。

假设 Bob 对消息 $m = 5$ 的签名 $(r, s) = (6, 3)$ 进行验证，因为 $3^6 \times 6^3 \equiv 2^5 \bmod 11$，验证通过，确认其为有效签名。

3.1.2 ElGamal 签名设计的机理

思考 3.1 后验思考：已知方案的情况下验证方案的安全性。

(1) 抵御选择性伪造的安全性。

① 敌手在没有签名私钥 x 的情况下，试图对给定的消息 m 伪造签名。敌手拥有的信息是 p，g 和 y。方法是首先随机选择一个值 r，然后试图找到相应的 s，满足目标方程 $y^r r^s = g^m \bmod p$，为达到这一企图，需要求出 $\log_r(g^m y^{-r})$，即 s。但这是困难的，因为这是解离散对数问题。

② 如果敌手首先选择 s，然后试图找到 r，则需要求解关于 r 的方程

$$y^r r^s \equiv g^m \bmod p$$

求解该等式目前为止没有可行的办法。

③ 如果敌手已知消息 m，试图同时求(r,s)，需要求关于 r，s 的方程

$$y^r r^s \equiv g^m \bmod p$$

目前仍没有人发现求解这一问题的方法，也没有人证明不能求解这个问题。

(2) 抵御存在性伪造的安全性。

如果敌手先选择 r 和 s，然后去解 m，企图构造一次存在性伪造攻击，那么需要求 $\log_g y^r r^s (=m)$，这又一次面临求解离散对数问题。

当同时选择 r，s，m，则存在性伪造是可能的(见 3.1.3 节中的安全性分析)。但是可通过先使用散列函数来排除这种可能。

思考 3.2 先验思考：方案是怎么想出来的？

必然有一个关键方程建立了签名与消息 m 之间的联系，即存在一个验证签名的等式，该等式中必然包含签名和消息 m。

回顾思考 3.1，从(1)中①②的分析来看，签名的两项(r 和 s)必须至少在指数位置出现。从(2)的分析来看，m 也必须出现在指数位置。

于是，验证签名等式必然是形如 $A^r B^s \equiv C^m \bmod p$，$A$，$B$，$C$ 可取的值为公开的信息 y，g，r，s。

然后，通过验证方程反推(构造)出签名方程。例如，假设取 $A=r$，$B=g$，$C=y$，则验证方程为 $r^r g^s = y^m \bmod p$。因为总是有 $r=g^k \bmod p$，于是有 $g^{kr} g^s = g^{mx}$，故 $kr+s=mx$，即 $s=mx-kr \bmod (p-1)$。这便得到了一个签名方程。

思考 3.3 一般性发散思维：该签名设计方案的一般性原理是什么？

参见 4.1 节，这里只是简要提及。

设计中的重要环节是签名过程，回顾该过程可以发现有两个函数，一个是选取随机变量构造一个 $r=g^k \bmod p$，该函数引入随机性 k；另一个函数的一般模型是 $ak=b+cx \bmod (p-1)$，这里 a，b，c 可以是 m，r，s 的函数。计算出 s，可见 s 中蕴含了签名者对 m，x，k 的拥有。签名就是(r,s)。

更深入地解释需要用到实体认证(entity authentication)和身份识别(identification)协议的概念，如零知识(zero knowledge)、比特承诺(bit commitment)。可以参见第 5 章，这里只作简要介绍。签名协议可视为“承诺(证据)—挑战(challenge)—应答(response)”机制，需要交互双方共同配合，采取“挑战应答”的形式。“应答”的验证成功表明签名者知道签名密钥(私钥)x。为了将 x 传递到签名验证者，需要将 x 进行“掩盖”，“掩盖”的方法是利用随机数 k。由于 k 是签名者随机选取的，需要将关于 k 的信息“同时”传递给签名

验证方，这就是"承诺"，即承诺一个随机数 k。在签名过程中，由于没有交互过程，"挑战"环节由签名者自己完成，但要保证"挑战"的随机性。利用随机数 k 计算 $r=g^k \bmod p$，r 作为"承诺"，"承诺"(r)本身的单向性表现出一定随机性，可视为随机"挑战"。签名方程中 $ak=b+cx \bmod (p-1)$ 蕴含着"承诺"r、"应答"s 和私钥 x 之间的关系(即 a,b,c 为 m,r,s 的组合)。利用签名方程 $ak=b+cx \bmod (p-1)$ 计算变量 s，s 便视为"应答"。"应答"s 表明对随机"挑战"的回答，计算"应答"s 需要用到签名私钥 x，故"应答"s 表明了对私钥 x 的拥有，另外，由于 k 的随机性保证了"应答"中没有透露私钥 x 的信息，成功地"掩盖"了 x。

更一般性的解释见 5.3 节。

3.1.3 安全性分析、性能分析与比较

1. 安全性分析

(1) 整体性攻击。ElGamal 的安全性基础是离散对数问题的困难性，所以 p 必须是一个充分大的素数，否则利用穷举搜索法可以很容易地由已知(p,g,y)从同余方程 $y=g^x \bmod p$ 算出私钥 x。因此，p 至少应该是二进制 512 位的素数(避免指数演算法攻击)，从长期安全考虑，推荐使用 1024 位或者更长的密钥。另外，$p-1$ 最好有大的素数因子(避免 Pohlig-Hellman 算法攻击)，私钥 x 最好是 $\mathbb{Z}_p^*$ 的素数阶子群的生成元。

(2) 存在性伪造。一种是唯密钥攻击条件下的存在性伪造。伪造者可任选两数 u,v：$1\leqslant u,v<p-1$，$\gcd(v,p-1)=1$，计算

$$r = g^{-u}y^v \bmod p$$
$$s = -rv^{-1} \bmod (p-1)$$
$$m = -us \bmod (p-1)$$

可以验证 $y^r r^s = g^{xr}(g^{-u}y^v)^{-rv^{-1}} = g^{xr+urv^{-1}}y^{-r} = g^{urv^{-1}} = g^m \bmod p$ 成立，所以(r,s)是对消息 m 的有效签名。由于这样的消息 m 不可事先确定，因此 m 可能毫无意义。这是唯密钥攻击条件下的存在性伪造。

另一种是已知消息攻击条件下的存在性伪造。假定(r,s)是消息 m 的有效签名，那么敌手可以利用这个"消息-签名"对伪造其他消息的签名。设 h,i,j 是整数，$0\leqslant h,i,j\leqslant p-2$，且 $\gcd(hr-js,p-1)=1$，计算

$$\lambda = r^h\alpha^i\beta^j \bmod p$$
$$\mu = s\lambda(hr-js)^{-1} \bmod (p-1)$$
$$m' = \lambda(hx+is)(hr-js)^{-1} \bmod (p-1)$$

容易验证，可以得到一个对 m' 的有效签名(λ,μ)。

(3) 随机数 k 要秘密选取，不可泄露，否则根据 k 可求得签名私钥 $x=r^{-1}(m-sk) \bmod (p-1)$(如果有 $\gcd(r,p-1)=1$)。

(4) 随机数 k 不能重复使用，否则有可能受到重复性攻击。例如，用 k 对消息 m_1 和 m_2 分别产生了两个签名(r,s_1)和(r,s_2)，则可通过解关于未知数 x 与 k 的同余方程组

$$\begin{cases} m_1 = rx + s_1k \bmod (p-1) \\ m_2 = rx + s_2k \bmod (p-1) \end{cases}$$

求得签名私钥 x。即 $m_1-m_2=k(s_1-s_2) \bmod (p-1)$，设 $d=\gcd(s_1-s_2, p-1)$，因为 $d|p-1$，且 $d|(s_1-s_2)$，可证明 $d|(m_1-m_2)$，定义 $m'=\frac{m_1-m_2}{d}$，$s'=\frac{s_1-s_2}{d}$，$p'=\frac{p-1}{d}$，则有 $m'=ks' \bmod p'$，且 $\gcd(s', p')=1$，那么 $k=m'(s')^{-1} \bmod p'$，$k=tp'+m'(s')^{-1}(0\leqslant t\leqslant d-1)$，通过测试 $r=g^k \bmod p$ 来确定哪一个是 k。

(5) 签名者多次签名时选取的多个 k 之间应无关联，否则可能受到同态性攻击。例如，3 个不同的签名所选取的随机数为 k_1, k_2, k_3，满足 $k_3=k_1+k_2$，故由 $s\equiv(m-xr)k^{-1} \bmod (p-1)$ 可得 $m\equiv ks+xr \bmod (p-1)$。于是

$$m_1 \equiv (xr_1 + k_1 s_1) \bmod (p-1)$$
$$m_2 \equiv (xr_2 + k_2 s_2) \bmod (p-1)$$
$$m_3 \equiv (xr_3 + k_3 s_3) \bmod (p-1)$$

对以上三式分别乘以 s_2s_3, s_1s_3, s_1s_2，得：

$$m_1 s_2 s_3 \equiv xr_1 s_2 s_3 + k_1 s_1 s_2 s_3 \bmod (p-1) \tag{1}$$
$$m_2 s_1 s_3 \equiv xr_2 s_1 s_3 + k_2 s_1 s_2 s_3 \bmod (p-1) \tag{2}$$
$$m_3 s_1 s_2 \equiv xr_3 s_1 s_2 + k_3 s_1 s_2 s_3 \bmod (p-1) \tag{3}$$

式(1)＋式(2)－式(3)，利用 $k_3=k_1+k_2$，可求出签名私钥：

$$x \equiv (m_1 s_2 s_3 + m_2 s_1 s_3 - m_3 s_1 s_3)(r_1 s_2 s_3 + r_2 s_1 s_3 - r_3 s_1 s_2)^{-1} \bmod (p-1)$$

(6) 代换攻击。设 (r, s) 是 m 的一个签名，若 r 可逆，设 $k'=tk+n$，t, n 为任意两个整数，且满足 $k'\in\mathbb{Z}_{p-1}$，$r'=r^t g^n \bmod p$，$s'=skr^{t-1}g^n(tk+n)^{-1} \bmod (p-1)$，则 (r', s') 是 m' 的签名，其中 $m'=r^{t-1}g^n m \bmod (p-1)$。从而完成已知消息攻击条件下的存在性伪造。

理由如下：(r, s) 是 m 的一个签名，则有 $m=(xr+sk) \bmod (p-1)$，若 r 可逆，又有 $x=r^{-1}(m-sk) \bmod (p-1)$，由签名方程得 $m'=s'k'+xr' \bmod (p-1)$，将 s', k', x, r' 代入方程得 $m'=(r^{t-1}g^n m) \bmod (p-1)$。

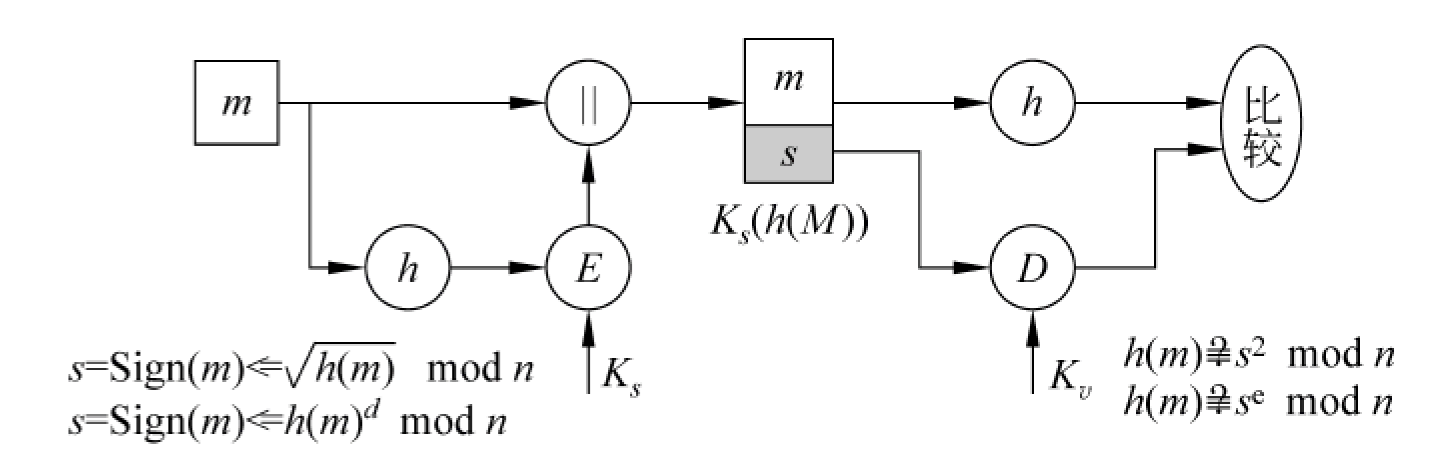
由(3)、(4)和(5)可知，随机数 k 的选取和保管对私钥 x 的保密性起重要的作用。3 种攻击的目标都是完全攻破。问题(2)和(5)可通过在签名之前先使用散列函数来解决。

2. 性能分析

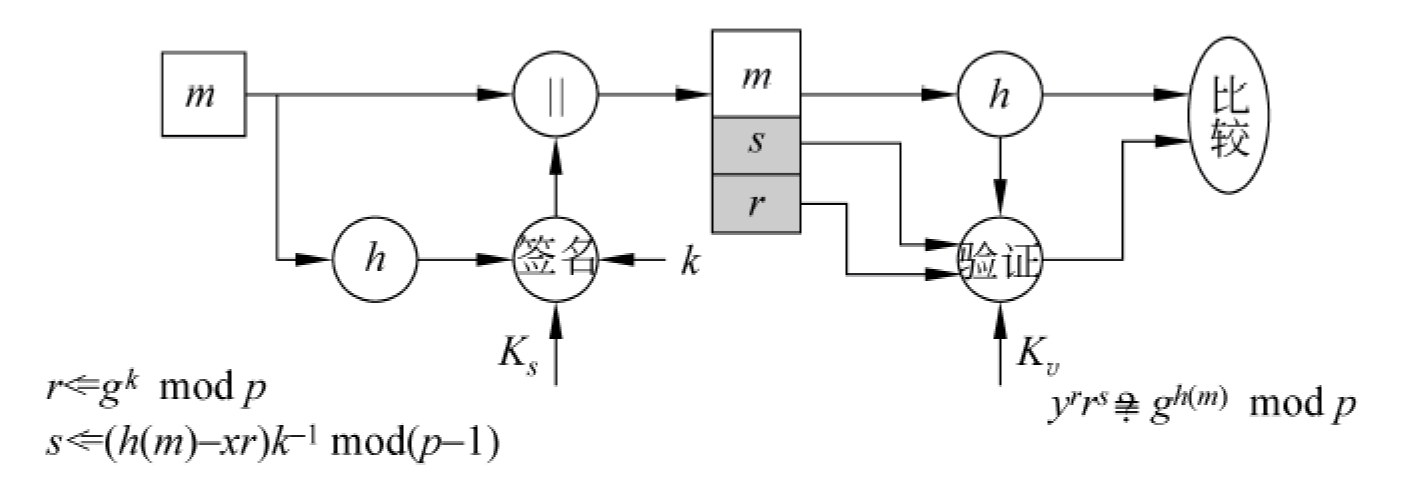
ElGamal 签名生成需要做一次模指数运算($r=g^k \bmod p$)、一次扩展的 Euclidean 算法运算(求随机数 k 的逆元)和二次模乘运算($s=(m-xr)k^{-1} \bmod (p-1)$)，前两个运算可以离线进行。

3. Rabin、RSA 和 ElGamal 在签名设计思路上的比较

Rabin、RSA 是采用(类似于加密体制中的)解密函数进行签名，用加密函数进行验证，签名只由一项构成。而 ElGamal 签名是构造了一种"等式关系"进行签名，因此，后者的签名中拥有两个部分(一个是"真正"的签名部分 s，一个是"承诺"r)。这种"等式关系"证明了签名者对私钥的拥有。因此，如何构造含有签名、消息和私钥等信息的"等式关系"成为设计签名方案的关键，如图 3.1 所示。



(a) Rabin、RSA签名体制



(b) ElGamal签名体制

图 3.1 Rabin、RSA 和 ElGamal 签名在设计思路上的差异

3.2 Schnorr 签名

C. Schnorr 在 1989 年提出了该数字签名方案，该方案具有签名速度较快，签名长度较短等特点。下面简要介绍其方案的实现过程。

（1）密钥生成：首先选择两个大素数 p 和 q，q 是 $p-1$ 的大素数因子，然后选择一个生成元 $g\in\mathbb{Z}_p^*$，且 $g^q\equiv1 \bmod p$，$g\neq1$，最后选择随机数 x，$1<x<q$，计算 $y=g^x \bmod p$，则公钥为(p,q,g,y)，私钥为 x。

（2）签名过程：签名者选择随机数 k，$1\leqslant k\leqslant q-1$，然后计算：

$$r = g^k \bmod p$$
$$e = h(m,r)$$
$$s = (xe+k) \bmod q$$

得到的签名为(e,s)，其中 h 为安全的散列函数，$h:\{0,1\}^*\to\mathbb{Z}_p$。

（3）验证过程：签名接收者在收到消息 m 和签名(e,s)后，首先计算

$$r' = g^s y^{-e} \bmod p$$

然后验证等式

$$e \overset{?}{=} h(m,r')$$

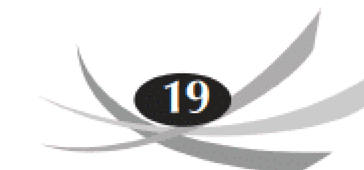
如等式成立，则签名有效；否则，签名无效。

正确性证明

因为

$$r = g^k \bmod p,\ e = h(m,r),\ s = (xe+k) \bmod q$$

故



$$r' = g^s y^{-e} \bmod p = g^s g^{-xe} \bmod p = g^{s-xe} \bmod p = g^{xe+k-xe} \bmod p$$
$$= g^k \bmod p = r$$

因此

$$h(m, r') = h(m, r) = e$$

例 3.2 一个 Schnorr 签名的实际例子。

(1) 密钥生成：Alice 选择素数 $p=129\ 841$ 和 $q=541$，这里 $(p-1)/q=240$。然后 Alice 选取随机数 $g=26\ 346 \in \mathbb{Z}_p^*$，并计算 $g=26\ 346^{240} \bmod p=26$。既然 $g \neq 1$，那么 g 生成 $\mathbb{Z}_p^*$ 中唯一的 541 阶循环子群。Alice 接着选取私钥 $x=423$，计算 $y=26^{423} \bmod p=115\ 917$。则 Alice 的公钥是 $(p=129\ 841, q=541, g=26, y=115\ 917)$。

(2) 签名生成：为签署消息 $m=11101101$，Alice 选取随机数 $k=327$，$1 \leqslant k \leqslant 540$，并计算 $r=26^{327} \bmod p=49\ 375$，$e=h(m,r)=155$。最后 Alice 计算 $s=423 \times 155+327 \bmod 541=431$。从而签名为 $(s=431, e=155)$。

(3) 签名验证：Bob 计算 $r'=26^{431} \times 115\ 917^{-155} \bmod p=49\ 375$，$h(m,r')=155=e$。故 Bob 接收该签名。

性能分析

签名生成需要模 p 的一次指数运算 $(r=g^k \bmod p)$ 和模 q 的一次乘法运算 $(s=(xe+k) \bmod q)$。模 p 指数可以离线计算。计算 $h(m,r)$ 的时间应相对较少，具体与散列函数的选择有关。验证过程需要模 p 的两次指数运算。采用 q 阶子群并没有对 ElGamal 方案有很大的改进，但确实比 ElGamal 方法有更短的签名，因为 e 和 s 都比 p 短。

安全性分析

Schnorr 数字签名方案的参数选取与 ElGamal 数字签名方案不同，ElGamal 方案中 g 为 $\mathbb{Z}_p^*$ 的生成元，而在 Schnorr 数字签名方案中，g 为 $\mathbb{Z}_p^*$ 的 q 阶子群的生成元。从穷尽搜索签名者私钥的角度而言，ElGamal 签名的安全性较高，其生成元的阶较大。此外，两者安全性相似。

思考 3.4 Schnorr 数字签名的设计机理。

Schnorr 数字签名是典型的基于身份识别协议的转换形式，通过非交互零知识证明协议，证明了对签名私钥的拥有。还是基于"承诺—挑战—应答"三步骤，$r=g^k \bmod p$ 是对 k 的"承诺"，其随机"挑战"十分典型，为 $h(m,r)$，"应答"则是 $s=(xe+k) \bmod q$，从而证明了对私钥的拥有，并通过 k 的随机性"掩盖"了 x 的传递，不泄露关于 x 的信息。

3.3 数字签名标准 DSS

1994 年 12 月美国 NIST 正式颁布了数字签名标准 DSS(FIPS 186)。它是在 ElGamal 和 Schnorr 数字签名方案基础上设计的。DSS 最初建议使用 p 为 512 位的素数，q 为160 位的素数，后来在众多批评下，NIST 将 DSS 的密钥 p 从原来的 512 位增加到 512～1024 位。当 p 选为 512 位的素数时，ElGamal 签名的长度为 1024 位，而 DSS 中通过 160 位的素数 q 可将签名的长度降低为 320 位，这就减少了存储空间和传输带宽。DSS 中的算法称为 DSA(Digtial Signature Algorithm，数字签名算法)。

DSA 的描述如下：

(1) 密钥生成：选取一个素数 p：$2^{L-1}<p<2^L$，L 为 64 的倍数。

选取 $p-1$ 的一个素数因子 q：$2^{159}<q<2^{160}$。

取 $g=\alpha^{(p-1)/q} \bmod p$，其中 α 是使 $1<\alpha<p-1$ 及 $\alpha^{(p-1)/q} \bmod p>1$ 成立的整数。随机选取整数 x：$0<x<q$。

计算 $y=g^x \bmod p$。

选取安全的散列函数 h：对消息 m，$h(m)$是 160 位的消息摘要。

(p,q,g)是公开参数，x，y 分别是签名者的私钥和公钥。

(2) 签名生成：对消息 $m\in\mathbb{Z}_p^*$，Alice 随机选取一个整数 k：$1\leqslant k<q$，并计算

$$r = (g^k \bmod p) \bmod q$$

$$s = k^{-1}(h(m) + xr) \bmod q$$

(r,s)是 Alice 对消息 m 的签名。将(r,s)发送给 Bob(如果 $r=0$ 或 $s=0$，则选取新的随机数 k，重新计算出 r 和 s。一般来说，$r=0$ 或者 $s=0$ 的出现概率极小)。

(3) 签名验证：Bob 收到(r,s)后，先检验 $0<r<q$，$0<s<q$ 是否成立。如果有一个不成立，则(r,s)不是 Alice 的签名。如果两者成立，则用 Alice 的公钥 y 及公开信息(p,q,g)，计算

$$w = s^{-1} \bmod q$$

$$u_1 = h(m)w \bmod q$$

$$u_2 = rw \bmod q$$

$$v = ((g^{u_1} y^{u_2}) \bmod p) \bmod q$$

如果 $v=r$，则 Bob 接受(r,s)是 Alice 对消息 m 的有效签名。否则，拒绝此签名。

正确性证明

$$\begin{aligned} v &= ((g^{u_1} y^{u_2} \bmod p) \bmod q) = ((g^{u_1} (g^x)^{u_2} \bmod p) \bmod q \\ &= (g^{u_1+xu_2} \bmod p) \bmod q = (g^{h(m)w+xrw} \bmod p) \bmod q \\ &= (g^{skw} \bmod p) \bmod q = (g^{sks^{-1}} \bmod p) \bmod q \\ &= (g^k \bmod p) \bmod q = r \end{aligned}$$

验证过程写成 4 步的目的是清楚地表明签名验证需要求逆、求积(3 次)、求幂(2 次)。在 q 阶子群中运算的好处是签名长度从原来的 $2p$ 减少到 $2q$。

DSA 签名方案看上去较为复杂，这里给出一个图示便于从全局上理解(见图 3.2，签名验证的函数分开写是为了说明验证的计算量)。

例 3.3 取 $q=101$，$p=78q+1=7879$，3 是$\mathbb{Z}_{7879}^*$中的一个本原元，取 $g=3^{78} \bmod 7879=170$，显然 g 是 1 模 p 的q 次根。假设 $x=75$，那么 $y=g^x=170^{75} \bmod 7879=4567$。假设 Alice 要计算对消息摘要 SHA1$(m)=22$ 的签名。

(1) 签名过程：计算

$k^{-1} \bmod 101 = 50^{-1} \bmod 101 = 99$

$r = (g^k \bmod p) \bmod q = (1750^{50} \bmod 7879) \bmod 101 = 2518 \bmod 101 = 94$

$s = (h(m) + xr)k^{-1} = (22 + 75 \times 94)99 \bmod 101 = 97$

对消息摘要 22 的签名为(94,97)。

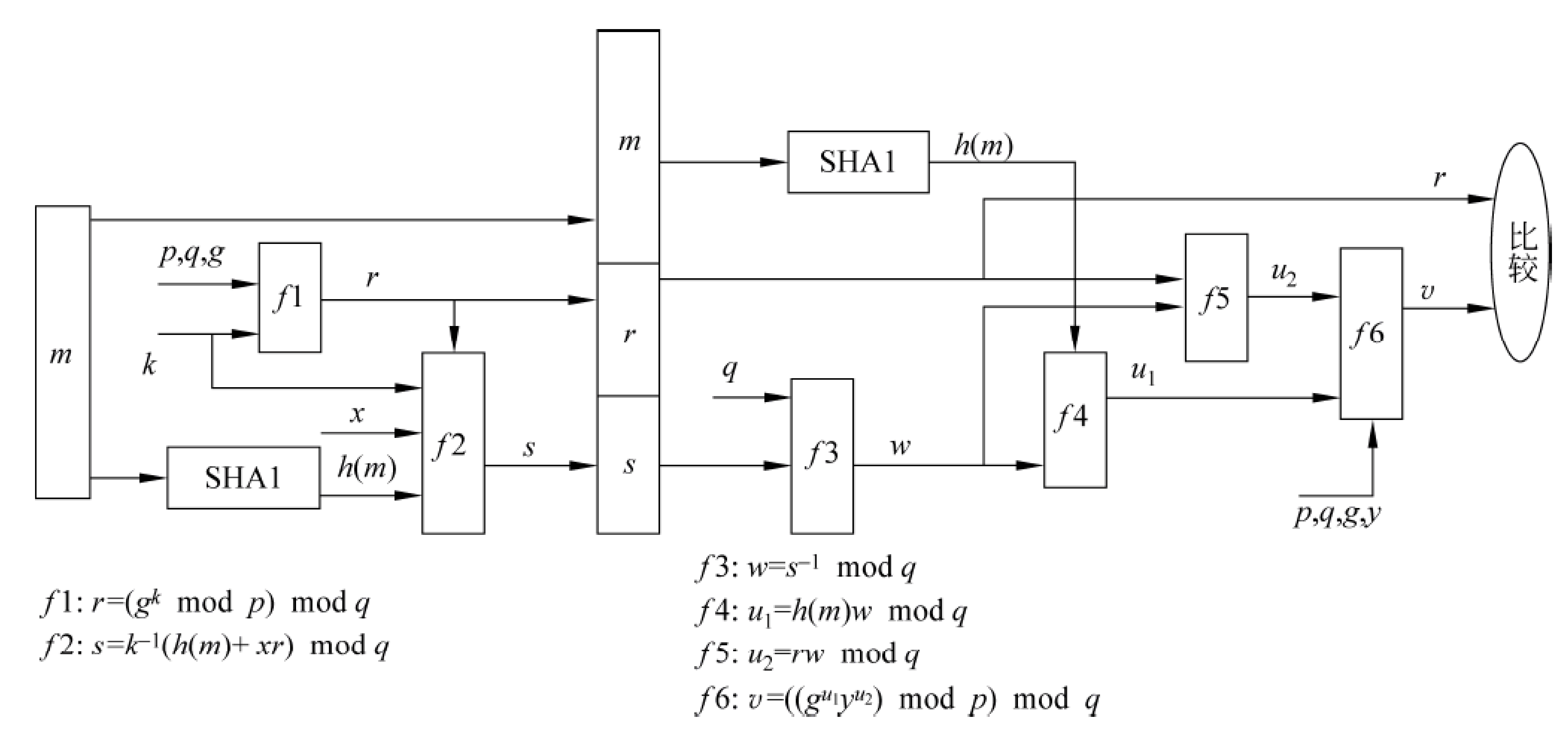


图 3.2　DSA 数字签名算法的框图

(2) 签名验证：

$$s^{-1}=97^{-1} \bmod 101=25$$

$$u_1=22\times 25 \bmod 101=45$$

$$u_2=94\times 25 \bmod 101=27$$

$$v=(170^{45}\times 4567^{27} \bmod 7879) \bmod 101=2518 \bmod 101=94=r$$

签名有效。

1. 安全性讨论

(1) DSS 数字签名标准的安全性基于两个离散对数问题：一是基于乘法群$\mathbb{Z}_p^*$上的离散对数问题，二是基于其 q 阶子群上的离散对数问题。已知的最好的攻击是亚指数算法。

(2) DSS 并未很明确地指明 k 值选取的随机性，其实 k 的选取的基本要求和 ElGamal 签名方案中的类似。

(3) DSS 中使用的散列函数一般采用安全散列函数标准 FIPS PUB180 中公布的 SHA1。

2. 基于离散对数问题的签名体制 ElGamal、Schnorr、DSA 的比较

从方案提出的时间来看，ElGamal 签名方案是最先提出的，是后两种方案的基础。第二个提出的方案是 Schnorr 签名方案，是 ElGamal 签名方案的一种变体，主要目的是缩短签名长度。DSA 是 ElGamal 方案的另一种变体，并吸收了 Schnorr 签名方案的设计思想，如缩短签名的长度。下面具体介绍 3 种签名方案的联系与区别。

(1) 从参数的初始化上可以看到，DSA 和 Schnorr 方案中通过引入素数 q 并选择$\mathbb{Z}_p^*$的 q 阶子群的生成元，修改了 ElGamal 方案中直接选择$\mathbb{Z}_p^*$本身的生成元的做法。使得方案的安全性依赖于两个不同的但又相关的离散对数问题，即$\mathbb{Z}_p^*$上的离散对数问题和$\mathbb{Z}_q^*$上的离散对数问题。这两种方案的签名长度较 ElGamal 方案短。

(2) 签名过程中 Schnorr 方案所用的散列函数不只是消息 m 的函数，还是 r 的函数，

这一点与另外两个方案不同。

(3) 作为标准,DSA 中限制了 q 的长度为 160 位,p 的长度为 512～1024 位之间 64 的任何倍数,并规定了散列算法是 SHA1。其他签名方案中没有这些具体的限制。

由于签名方程有区别,故验证方程也有差别。从性能分析角度比较 3 种方案的效率,性能指标是计算量和签名长度。考察的运算主要是求幂(T_E)、乘积(T_M)、散列函数求值(T_H),而计算量相对较小的运算如模加、模减、求逆运算的所耗时间忽略不计。图 3.3 给出 3 种方案的总结图。效率比较结果见表 3.1。

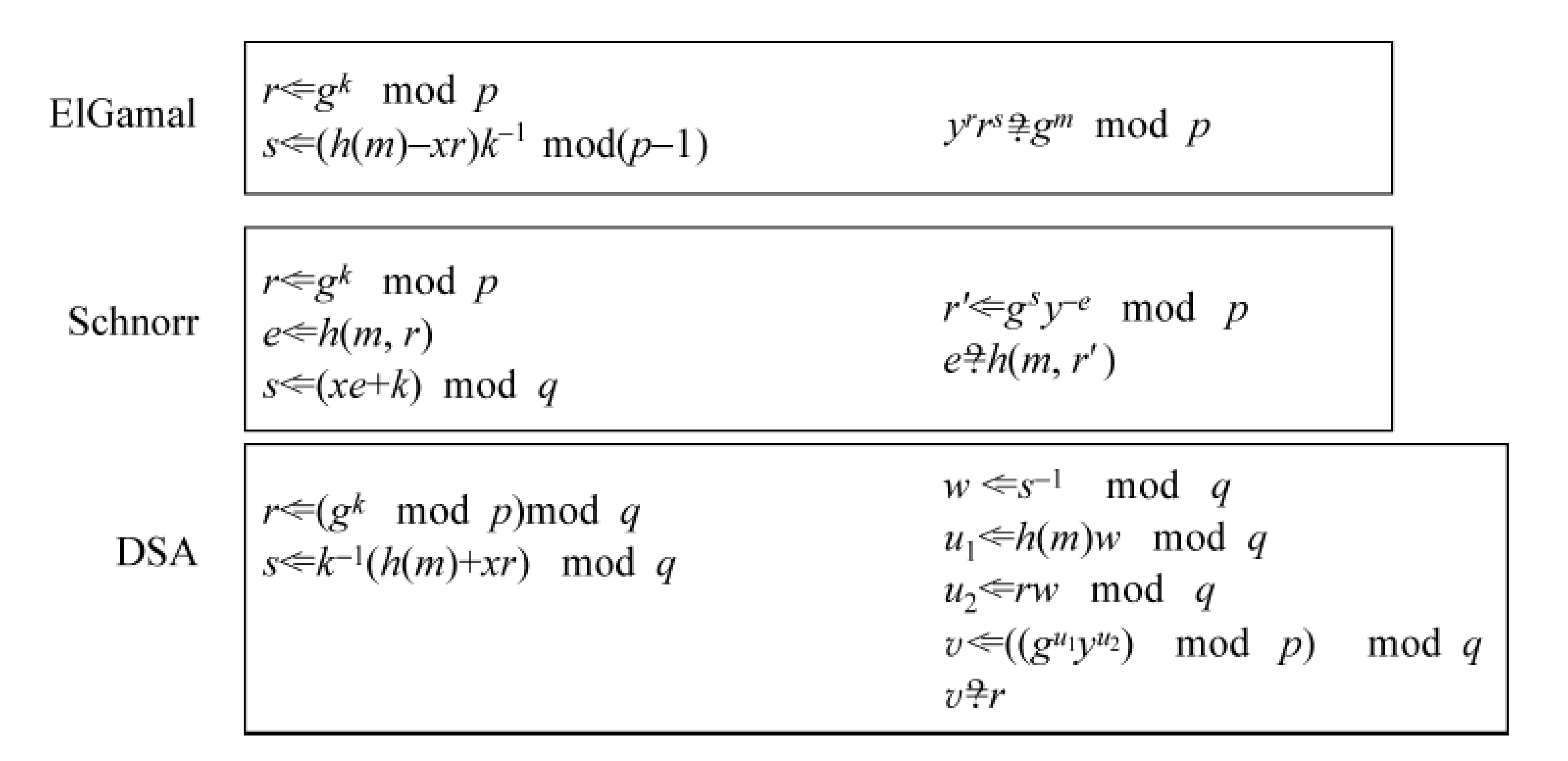


图 3.3　3 种离散对数签名方案的总结

表 3.1　3 种离散对数签名方案的性能比较

签名体制	签　　名	验　　证	签 名 长 度
ElGamal	$T_E+T_H+2T_M$	$3T_E+T_H+T_M$	$\|p\|+\|p-1\|$
Schnorr	$T_E+T_H+T_M$	$2T_E+T_H+T_M$	$\|q\|+\|h()\|$
DSA	$T_E+T_H+2T_M$	$2T_E+T_H+3T_M$	$2\|q\|$

可见,Schnorr 方案的签名过程计算量相对较小,速度较快,尤其有些计算与消息无关,可以预先完成,这也能够减小签名的时间。对于验证过程,同样是 Schnorr 方案的计算量相对较小。并且,Schnorr 方案的签名长度也较短。因此,该方案较适合在智能卡等环境应用。其他两种方案的个别计算,如求随机数的幂 r、求 xr,可以预先计算。

3.4　Neberg-Rueppel 签名

前面介绍的 3 种签名方案可视为 ElGamal 方案及其变体,它们都是带消息的签名方案,即消息作为签名验证方程的输入。1994 年,Neberg 和 Rueppel 提出一种具有消息恢复功能的签名方案,且方案中在签名产生和签名验证方程中无须求逆。验证者可以从签名中恢复出原始消息,因此签名者不需要将被签消息发送给验证者。

(1) 参数产生:p 是一个大素数;q 是一个大素数,$q\mid(p-1)$。$g\in\mathbb{Z}_p^*$,随机选取,且 $g^q\equiv 1 \bmod p$。私钥为 $x\in\mathbb{Z}_p^*$,公钥为 $y\Leftarrow g^x \bmod p$。

(2) 签名产生:计算 $\tilde{m}=R(m)$,这里 R 是一个单一映射,且容易求逆,称为冗余

函数。

选择一个随机数 $k(0<k<q)$，计算

$$r \Leftarrow g^{-k} \bmod p$$
$$e \Leftarrow \widetilde{m} r \bmod p$$
$$s \Leftarrow xe + k \bmod q$$

得到消息 m 的签名为 (e,s)。

(3) 签名验证：验证是否 $0<e<p$，$0\leqslant s<q$。计算

$$v \Leftarrow g^{s} y^{-e} \bmod p$$
$$m' \Leftarrow ve \bmod p$$

如果 $m' \in M_R$，M_R 表示 R 的值域，恢复 $m=R^{-1}(m')$。否则拒绝该签名。

正确性证明

$$m' = ve \bmod p = g^{s} y^{-e} e \bmod p = g^{xe+k-xe} \bmod p = g^{k} e \bmod p = \widetilde{m}$$

冗余函数可取 $R(m)=\{m \parallel m\}$，即简单地将原消息复制后链接。

思考 3.5　该方案的一般设计机理。

$e=\widetilde{m} r \bmod p$ 其实可视为一个简单的加密，密钥为 r。签名为加密的消息 e、私钥 x 以及随机数 k 的运算组成。验证签名时先计算密钥 $v(=r)$，然后解密来恢复消息 $m'=ve \bmod p$。这可推广为任何对称密钥算法。令 $E=\{E_r: r\in \mathbb{Z}_p\}$ 为加密变换的集合，其中 E_r 为以密钥 $r\in \mathbb{Z}_p^*$ 为索引的加密变换，且是从 $\mathbb{Z}_p^*$ 到 $\mathbb{Z}_p^*$ 的双射。对任意 $m\in M$，选择随机整数 k，$1\leqslant k\leqslant q-1$，计算 $r=g^{k} \bmod p$，$e=E_r(m)$，$s=xe+k \bmod q$，则 (e,s) 是消息 m 的签名。

第 4 章

离散对数签名的扩展*

4.1 基于离散对数的一般签名

总结第 3 章介绍的 4 种签名方式，可以得出基于离散对数问题的一般签名方案。

(1) 参数产生。选取一个满足安全性要求的大素数 p，q 为 $p-1$ 的大素数因子。选取 $g\in\mathbb{Z}_p^*$，且 $g^q\equiv 1 \bmod p$。选取随机数 $x(1<x<q)$，作为私钥。计算 $y\equiv g^x \bmod p$，(p,q,g,y)为公钥。

(2) 签名产生。计算 $h(m)$，h 为安全的散列函数。选择随机数 k，满足 $1<k<q$，计算 $r\equiv g^k \bmod p$；从签名方程 $ak\equiv b+cx \bmod q$ 解出 s。这里(a,b,c)有许多种不同的选择方法，如表 4.1 所示。

表 4.1 (a,b,c)的可能置换

$\pm r$	$\pm s$	m	$\pm mr$	$\pm sr$	1
$\pm mr$	$\pm s$	1	$\pm ms$	$\pm sr$	1
$\pm mr$	$\pm ms$	1			

其中(a,b,c)可取表中某一行的 3 个值的任意排列。

(3) 验证过程。设验证方程为 Vrfy，签名接收者收到消息(r,s)后，可以按照以下验证方程验证签名的有效性：$\mathrm{Vrfy}(y,(r,s),m)=\mathrm{True}\Leftrightarrow r^a\equiv g^b y^c \bmod p$。以表 4.1 的第一行为例，忽略负号，$(a,b,c)$的可能值为$(r,s,m)$的置换，故为 $3!=6$ 种情况，如表 4.2 所示。

表 4.2 签名方程($ak\equiv b+cx \bmod q$)和相应的验证方程

(a,b,c)	签 名 方 程	验 证 方 程
(r,s,m)	$rk\equiv s+mx \bmod q$	$r^r\equiv g^s y^m \bmod p$
(r,m,s)	$rk\equiv m+sx \bmod q$	$r^r\equiv g^m y^s \bmod p$
(s,r,m)	$sk\equiv r+mx \bmod q$	$r^s\equiv g^r y^m \bmod p$
(s,m,r)	$sk\equiv m+rx \bmod q$	$r^s\equiv g^m y^r \bmod p$
(m,s,r)	$mk\equiv s+rx \bmod q$	$r^m\equiv g^s y^r \bmod p$
(m,r,s)	$mk\equiv r+sx \bmod q$	$r^m\equiv g^r y^s \bmod p$

如果考虑负号的情况，则为 6×4＝24 种情况。如果将表 4.1 所列的 5 种情况都考虑进去，则共有 24×5＝120 种签名方程，即 120 种签名产生的方式。

有几点需要说明：

(1) a,b,c 是关于自变量 r,s,m 的函数，且自变量的最高次数为 1。

(2) r,s,m 至少在 a,b,c 中出现一次。

(3) 通用签名方程 $ak \equiv b+cx \bmod q$ 能解出 s。

思考 4.1　第 3 章介绍的 4 种离散对数签名方案对应的通用签名方程的系数。

ElGamal 签名为 $(a,b,c)=(s,m,-r)$，DSA 签名为 $(a,b,c)=(s,h(m),r)$，Schnorr 签名为 $(a,b,c)=(1,-s,h(m,r))$，Neberg-Rueppel 签名为 $(a,b,c)=(1,s,-rR(m) \bmod p)$。

为了更接近 DSA，如果把方程 $r \equiv g^k \bmod p$ 改为 $r \equiv (g^k \bmod p) \bmod q$，不改变签名方程 $ak \equiv b+cx \bmod q$，则验证方程变为 $(r \bmod q)^a = g^b y^c \bmod p$，这里

$$u_1 = a^{a-1} b \bmod q$$

$$u_2 = a^{a-1} c \bmod q$$

$$r = (g^{u_1} y^{u_2} \bmod p) \bmod q$$

这样，又可以构造 120 种方案。更多分析参见相关文献①。

4.2　一般签名方案的举例

4.2.1　GOST 签名

该算法是俄罗斯的数字签名标准，它于 1995 年启动，全称为 GOST R34.10—94。与美国标准 DSA 算法很相似，与之相比，GOST 算法使用的散列函数不是 SHA1，而是俄罗斯自己颁布的建立在 GOST 28174—89 分组密码算法基础上的 GOST 34.11—94。GOST 是 Gosudarsvennyi Standard 的缩写，泛指一系列俄罗斯密码标准。

(1) 参数产生。和 DSA 一样，系统产生全局参数、用户公钥和私钥。p 是一个大素数，长度为 509～512 位；q 是一个素数，且是 $p-1$ 的因子，长度为 254～256 位，p,q 都由标准给出的素数产生算法生成；g 是一个整数，满足条件 $g<p-1, g^q=1 \bmod p$。用户私钥为 $x, x<q$；用户公钥为 $y, y=g^x \bmod p$。

(2) 签名过程。

① 计算 m 的散列值 $H(m)$，散列函数由标准给出，散列值是一个长为 256 的比特串。如果 $H(m)=0$，那么重新设置散列值为 $H(m)=1$。

② 选取一个随机数 $k, 0<k<q$。

③计算 $r=(g^k \bmod p) \bmod q$，如果 $r=0$，返回②重新选取 k。

④ 计算 $s=(xr+kH(m)) \bmod q$，如果 $s=0$，返回②重新选取 k。

(r,s)为签名。

---

① P Horster, H Petersen, M Michels. Mete-ElGamal Signature Schemes. Proc. of ACM CCS94, pp. 96-107.

(3) 签名验证。首先检查 r 和 s 是否属于$[0,q]$,若不是,则(r,s)不是有效签名;判断 $g^s=y^r r^{H(m)} \bmod p$ 是否成立,若成立,则(r,s)为合法签名。

思考 4.2 GOST 签名对应 4.1 节中离散对数签名方案对应的通用签名方程。

$ak \equiv b+cx \bmod q$ 的系数是$(a,b,c)=(H(m),s,-r)$。

4.2.2 Okamoto 签名

该方案是 1992 年日本的 Okamoto 提出的一个签名方案。其设计的扩展思维在于它在 Schnorr 签名的基础上选取了两个随机整数,好处是可以达到可证明安全性。

(1) 参数产生。系统参数 $p\geqslant 2^{512}$,大素数 $q|(p-1)$且 $q\geqslant 2^{140}$;产生两个与 q 同长度的随机整数 g_1,g_2。用户的私钥是随机产生的两个随机整数 $x_1,x_2<q$。用户的公钥是 $y=g_1^{-x_1}g_2^{-x_2} \bmod p$。

(2) 签名产生。对于消息 m,签名者选取两个小于 q 的随机数 $k_1,k_2\in\mathbb{Z}_p^*$,计算

$$
\begin{aligned}
r &= g_1^{k_1}g_2^{k_2} \bmod p\\
e &= h(r,m)\\
s_1 &= k_1+ex_1 \bmod q\\
s_2 &= k_2+ex_2 \bmod q
\end{aligned}
$$

则(e,s_1,s_2)作为 m 的签名。

(3) 验证过程。接收者在收到消息 m 和签名(e,s_1,s_2)后,计算 $e'=h(g_1^{s_1}g_2^{s_2}y^e \bmod p,m)$,比较 $e'=e$ 是否成立,若成立,则签名有效,否则无效。

思考 4.3 如何进行正确性验证?

$$g_1^{s_1}g_2^{s_2}y^e \bmod p = g_1^{k_1+ex_1}g_2^{k_2+ex_2}(g_1^{-x_1}g_2^{-x_2})^e \bmod p = g_1^{k_1}g_2^{k_2} \bmod p$$

4.3 椭圆曲线上离散对数的签名

4.3.1 ECDSA

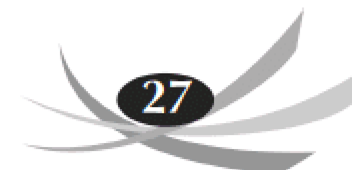
同加密方案一样,基于有限域离散对数问题的签名方案也可以移植到椭圆曲线上。数字签名方案 DSA 在椭圆曲线上的实现称为 ECDSA(Elliptic Curve Digtial Signature Algorithm),其困难性是基于有限域上椭圆曲线有理点群上离散对数问题的困难性。ECDSA 已于 1999 年被接受为 ANSI X9.62 标准,于 2000 年被接受为 IEEE P1363 以及 FIPS186-2 标准。

(1) 参数产生:设 GF(p)为有限域,E 是有限域 GF(p)上的椭圆曲线。选择 E 上的一点 $G\in E$,G 的阶为满足安全要求的素数 n,即 $nG=O$(O 为无限远点)。选择一个随机数 d,$d\in[1,n-1]$,计算 Q,使得 $Q=dG$,那么公钥为(n,Q),私钥为 d。

(2) 签名过程:

① 随机选择整数 k,$k\in[1,n-1]$,计算 $kG=(x,y)$,$r=x \bmod n$。

② 计算 $e=h(m)$,h 为安全散列函数。



③ 计算 $s=(e+rd)k^{-1} \bmod n$。如果 $r=0$ 或者 $s=0$，则另选随机数 k，重新执行上述过程。消息 m 的签名为 (r,s)。

(3) 验证过程：

① 计算 $e=h(m)$。

② 计算

$$
\begin{aligned}
u &= s^{-1}e \bmod n \\
v &= s^{-1}r \bmod n \\
(x',y') &= uG+vQ \\
r' &= x' \bmod n
\end{aligned}
$$

③ 如果 $r=r'$，则签名有效，否则无效。

正确性证明

由于

$$
\begin{aligned}
Q &= dG \\
s &= (e+rd)k^{-1} \bmod n \\
kG &= (x,y) \\
u &= s^{-1}e \bmod n \\
v &= s^{-1}r \bmod n \\
(x',y') &= uG+vQ
\end{aligned}
$$

故 $k\equiv(e+rd)s^{-1}\equiv(s^{-1}e+s^{-1}rd)\equiv(u+vd) \bmod n$，于是

$$(x,y)=kG=uG+vdG=uG+vQ=(x',y'),r'=x' \bmod n=x \bmod n=r$$

即 $r=r'$。

例 4.1　假设椭圆曲线为 Z_{23} 上的 $y^2=x^3+x+4$，参数分别为 $p=23$，$G=(0,2)$，$n=29$，$d=9$，$Q=dG=(4,7)$。

(1) 签名过程：Alice 签名，选取随机数 $k=3$，假设 $h(m)=4$，计算

$$
\begin{aligned}
(x,y) &= kG = 3(0,2) = (11,9) \\
r &= x \bmod n = 11 \bmod 29 = 11 \\
s &= (e+rd)k^{-1} \bmod n = (4+11\times 9)3^{-1} \bmod 29 = 15
\end{aligned}
$$

对 m 的签名为 $(11,15)$。

(2) 签名验证：Bob 接收到签名后计算

$$
\begin{aligned}
u &= s^{-1}e \bmod n = 15^{-1}\times 4 \bmod 29 = 8 \\
v &= s^{-1}r \bmod n = 15^{-1}\times 11 \bmod 29 = 22 \\
(x',y') &= uG+vQ = 8G+22Q = (11,9) \\
r' &= x' \bmod n = 11 \bmod 29 = 11 = r
\end{aligned}
$$

Bob 接受签名。

思考 4.4　能否给出基于椭圆曲线离散对数问题签名的一般版本？

ECDSA 与基于离散对数的一般签名的主要区别在于：在签名阶段，使用的是

$kG=(x,y)$的横坐标作为"承诺"。签名方程依然是形如 $ak\equiv b+cx \bmod q$,这里群的阶 q 改为 n,x 为私钥 d,$(a,b,c)=(s,h(m),r)$。验证方程不同,这是基于的困难问题不同导致的,用到的验证方程是$(x',y')=uG+vQ$,即这里用到的是椭圆曲线上的标量乘,而不是前面用到的有限域上的指数函数。

因此,一般地,对于签名方程 $ak\equiv b+cd \bmod n$,(a,b,c)为$(s,h(m),r)$的置换,则验证方程为$(x',y')=(a^{-1}bG+a^{-1}cdG)=(a^{-1}bG+a^{-1}cQ)$,然后比较横坐标。

对于 ECDSA 而言,$(a,b,c)=(s,h(m),r)$。基于离散对数或其他困难问题的签名几乎可以类比地"平移"到椭圆曲线签名体制上来,这一"平移"变换值得思考和体会。

安全性讨论

(1) 离散对数问题攻击。即通过解离散对数问题获得签名私钥。

(2) 如果 k 暴露,则敌手可获得签名私钥$(r^{-1}(ks-h(m)) \bmod n$。故 k 必须保密。

(3) 如果重复使用 k 用于 m_1,m_2 的签名,则由于 k 相同,故 r 相同,有

$$s_1k = h(m_1)+dr \bmod n,\quad s_2k = h(m_2)+dr \bmod n$$

于是 $k=(s_1-s_2)^{-1}(h(m_1)-h(m_2)) \bmod n$。

性能讨论

椭圆曲线密码具有密钥短,存储空间小,计算速度快,软硬件实现节省资源的特点。特别适用于计算能力和存储空间有限、带宽受限、要求高速实现的场合(如智能卡、传感器节点的应用中)。

4.3.2 SM2

中国国家密码管理局 2010 年 12 月 17 日发布 SM2 椭圆曲线公钥密码算法,该算法包含加密算法和签名算法。下面给出一个 SM2 签名算法的简单版本。

(1) 参数产生:设 GF(p)为有限域,E 是有限域 GF(p)上的椭圆曲线。选择 E 上的基点 $G\in E$,G 的阶为满足安全要求的 n,即 $nG=O$(O 为无限远点)。选择一个随机数 d,$d\in[1,n-1]$,计算 Q,使得 $Q=dG$,那么公钥为(n,Q),私钥为 d。

(2) 签名过程:

① 计算 $e=h(m)$,h 为安全散列函数。

② 随机选择整数 k,$k\in[1,n-1]$,计算 $kG=(x,y)$,$r=(e+x) \bmod n$。

③ 计算 $s=(1+d)^{-1}(k-rd) \bmod n$。

如果 $r=0$ 或者 $s=0$,则另选随机数 k,重新执行上述过程。消息 m 的签名为(r,s)。

(3) 验证过程:

① 计算 $e=h(m)$。

② 计算

$$t = (r+s) \bmod n$$
$$(x',y') = sG+tQ$$
$$r' = (e+x') \bmod n$$

③ 如果 $r=r'$,则签名有效,否则无效。

正确性证明

由于

$$\begin{aligned}(x', y') &= sG + tQ = sG + (r+s)dG = (s + rd + sd)G \\ &= (s(1+d) + rd)G = ((k - rd) + rd)G \\ &= kG = (x, y)\end{aligned}$$

于是有 $r = r'$。

第 5 章

基于身份识别协议的签名*

任何涉及“承诺(证据)—挑战—应答”交互序列的身份识别方案均可以转化为签名方案,即用证据 x 和要签署的消息 m 的连接,计算其单向散列函数值,即 $e=h(x \| m)$,该值代替验证者的随机挑战 e(这里 h 实质上起挑战者的作用)。将交互式身份识别方案转换为非交互数字签名方案时,挑战 e 的比特大小通常要增加到能排除对散列函数的离线攻击。

5.1　Feige-Fiat-Shamir 签名方案

Feige-Fiat-Shamir(FFS)方案根据 A. Fiat 和 A. Shamir 提出的 Fiat-Shamir 身份识别协议转换而来。FFS 数字签名的安全性基于模 n 平方根的困难性。

方案中需要使用单向散列函数 $h:\{0,1\}^* \to \{0,1\}^k$,其中 k 是给定的正整数。

方案描述如下:

(1) 密钥生成:

① 随机产生不同的秘密素数 p,q,并计算 $n=pq$。

② 选取正整数 k,及互不相同的随机整数 $s_1,s_2,\cdots,s_k \in \mathbb{Z}_n^*$。

③ 计算 $v_j=(s_j^2)^{-1} \bmod n, 1 \leqslant j \leqslant k$(即有 $v_j s_j^2=1 \bmod n, 1 \leqslant j \leqslant k$)。

A 的公钥是 k 维向量 $(v_1,v_2,\cdots,v_k)$ 和模数 n;A 的私钥是 k 维向量 $(s_1,s_2,\cdots,s_k)$。

(2) 签名生成:实体 A 执行如下操作。

① 随机选择一个整数 $r, 1 \leqslant r \leqslant n-1$。

② 计算 $u=r^2 \bmod n$。

③ 计算 $e=(e_1,e_2,\cdots,e_k)=h(m \| u)$,其中 $e_i \in \{0,1\}$。

④ 计算 $s = r\prod_{j=1}^{k} s_j^{e_j} \bmod n$。

A 对 m 的签名是 (e,s)。

(3) 验证签名:实体 B 验证 A 对 m 的签名 (e,s)。

① 获得 A 的可信公钥 $(v_1,v_2,\cdots,v_k)$ 和 n。

② 计算 $u' = s^2\prod_{j=1}^{k} v_j^{e_j} \bmod n$,以及 $e' = h(m \| u')$。

③ 当且仅当 $e=e'$ 时接受签名。

正确性证明

$$u' \equiv s^2\prod_{j=1}^{k} v_j^{e_j} \equiv r^2\prod_{j=1}^{k} s_j^{2e_j}\prod_{j=1}^{k} v_j^{e_j} \equiv r^2\prod_{j=1}^{k}(s_j^2 v_j)^{e_j} \equiv r^2 \equiv u \bmod n$$

因此，$u'=u$，于是 $e=e'$。

例 5.1　设 $n=35$，$k=4$，用户 A 的 4 个私钥为 $(s_1,s_2,\cdots,s_k)=(3,4,9,8)$，公钥为 $(v_1,v_2,\cdots,v_k)=(3^{-2},4^{-2},9^{-2},8^{-2}) \bmod 35=(4,11,16,29)$。

取 $r=16$，计算 $u=r^2 \bmod 35=16^2 \bmod 35=11$。

为了简化，不妨设 $e=h(m \| u)=1011_2$，即 1011 是二进制表示。于是

$$s = r\prod_{j=1}^{k} s_j^{e_j} \bmod n = 16\times 3^1\times 4^0\times 9^1\times 8^1 \bmod 35 = 26$$

得到的签名为 $(e,s)=(1011_2,26)$。

验证签名如下：

由于

$$w^2 = s^2\prod_{j=1}^{4} v_j^{e_j} \bmod n = 26^2\times 4^1\times 11^0\times 16^1\times 29^1 \bmod 36 = 11 = u$$

故签名有效。

安全性讨论

与 RSA 签名方案不同的是，FFS 方案中所有实体可使用相同的整数 n。在这种情形下，需要一个可信第三方产生素数 p 和 q 以及每个实体的公钥和私钥。

性能讨论

与 RSA 签名方式相比较，FFS 签名方式的优势是速度快。如在 RSA 方案中，指数的比特长度为 768，则生成签名平均需要 768 次模平方运算和 384 次模乘运算。而 FFS 方案中，相同的模数，随机数 k 为 128，则签署一条消息平均需要 64 次模乘运算，少于 RSA 签名所需工作量的 6%。但是，如果公钥为 3，则 RSA 签名验证需要一次模乘运算，而 FFS 签名验证平均需要 64 次模乘运算。因此，对于需要快速签名生成且不限制密钥空间存储量的应用，FFS 方案可能比 RSA 签名更合适。

5.2　Guillou-Quisquater 签名方案

FFS 方案是基于陷门单向函数 $u=r^2 \bmod n$，Guillou-Quisquater(GQ)方案则是基于陷门单向函数 $r=k^e \bmod n$。

方案描述如下：

(1) 密钥生成。

① 随机选取两个不同的大素数 p,q，计算 $n=pq$。

② 选择整数 $e\in\{1,2,\cdots,n-1\}$，使得 $\gcd(e,(p-1)(q-1))=1$。

③ 随机选择整数 $v\in\mathbb{Z}_n^*$，$\gcd(v,n)=1$(v 可视为 A 的身份识别号，如身份证号等)。

④ 确定整数 $a\in\mathbb{Z}_n$，满足 $va^e\equiv 1 \bmod n$。

于是公钥为 (n,e,v)，私钥为 a。

(2) 签名产生。实体 A 执行如下过程：

① 随机选择一个整数 r，计算 $u=r^e \bmod n$。

② 计算 $l=h(m \| u)$。

③ 计算 $s=ra^l \bmod n$。

得到对消息 m 的签名 (s,l)。

(3) 签名验证：验证 A 对 m 的签名 (s,l)，B 执行如下操作：

① 获得 A 的可信公钥 (n,e,v)。

② 计算 $u'=s^e v^l \bmod n$，以及 $l'=h(m \| u')$。

③ 当且仅当 $l=l'$ 时接受签名。

正确性证明

由于 $u' \equiv s^e v^l \equiv (ra^l)^e v^l \equiv r^e (va^e)^l \equiv r^e \equiv u \bmod n$，因此 $u=u'$，故 $l=l'$。

思考 5.1 上述 GQ 签名方案能否变换为带消息恢复的签名方案？

GQ 签名方案可以变换为带消息恢复的签名方案。方法是改变 $l=mr \bmod n$。其他不变。这样，在验证时有 $u'=s^e v^l \equiv r^e a^{el} v^l \equiv r^e \equiv u \bmod n$。消息 m 可由 $lr^{-1} \bmod n$ 恢复。

总结 FFS 方案和 GQ 方案，会发现一个一般规律，即两者的设计方法可以进行类比，见表 5.1。

表 5.1 FFS 和 GQ 方案的类比

设计步骤	FFS	GQ
1. 密钥产生(利用陷门单向函数)	$v_j s_j^2=1 \bmod n, 1\leqslant j\leqslant k$($s_j$ 为私钥，v_j 为公钥，基于 Rabin 问题)	$va^e\equiv 1 \bmod n$(a 为私钥，v 为公钥，基于 RSA 问题)
2. 计算对 r 的承诺 u(再次利用陷门单向函数)	$u=r^2 \bmod n$	$u=r^e \bmod n$
3. 生成随机挑战(利用单向散列函数 h，以及消息 m 和承诺 u)	$e=(e_1,e_2,\cdots,e_k)=h(m \| u)$	$l=h(m \| u)$
4. 签名生成(利用已承诺的随机值 r"掩盖"私钥，证明对私钥的拥有)	$s=r\prod_{j=1}^{k} s_j^{e_j} \bmod n$	$s=ra^l \bmod n$
5. 签名验证(根据公钥和私钥关系代入，承诺与随机值之间的关系)	$u'=s^2\prod_{j=1}^{k} v_j^{e_j} \bmod n$	$u'=s^e v^l \bmod n$

思考 5.2 在 11.4 节中介绍的 Schnorr 身份识别协议和 11.5 节中的 Okamoto 身份识别协议如何转化为签名方案？

转化为 Schnorr 签名方案的方法见 3.2 节。

转化为 Okamoto 签名方案的方法见 4.2.2 节。

5.3 知识签名

建议本节与 13.2 节的零知识证明协议结合学习。零知识证明是指示者向验证者证明他知道某个秘密知识，而没有向验证者泄露有关秘密知识的任何有用信息。交互零知识证明可以通过散列函数转换成非交互证明或签名。知识签名的概念是 Camenisch 和 Stadler 于 1997 年首次提出的。知识签名是由签名者将自己知道某知识(私钥)的信息附在消息的签名之上，但不泄露任何相关知识的内容。知识签名的本质是一种非交互式的

零知识证明。这里简单介绍几种知识签名。假设有安全散列函数：$H:\{0,1\}^* \to \{0,1\}^k$。

定义 5.1　满足等式 $c=H(m \| y \| g \| g^s y^c)$ 的数组 (c,s)，即为关于消息 m 的，y 以 g 为底的离散对数的知识签名，表示为

$$\mathrm{SPK}\{\alpha: y = g^\alpha\}(m)$$

其中 α 表示签名者持有的秘密。

这样一个知识签名 (c,s) 只有在知道秘密 $x=\log_g y$ 的情况下才能生成，当知道 x 时，签名者随机选取 $r \in \mathbb{Z}_n^*$，然后计算

$$c = H(m \| y \| g \| g^r), \quad s = r - cx \bmod n$$

得到签名 (c,s)，能生成这样一个签名说明了签名者知道 y 以 g 为底的离散对数 x，不知道 x 的情况下任何人想伪造一个签名都必须能够解决离散对数问题，所以这样一个知识签名可以证明 y 有 $y=g^x$ 的形式，并且签名者知道关于 $y=g^x$ 的秘密值 x。

定义 5.2　满足等式 $c=H(m \| y \| g \| h \| g^{s_1} h^{s_2} y^c)$ 的数组 (c,s_1,s_2)，即为关于消息 m 的，y 以 g，h 为底的离散对数的知识签名，表示为

$$\mathrm{SPK}\{(\alpha,\beta): y = g^\alpha h^\beta\}(m)$$

只有在知道满足等式 $y=g^{s_1} h^{s_2}$ 的秘密 (x_1,x_2) 时才能生成一个知识签名 (c,s_1,s_2)，当知道 (x_1,x_2) 的值时，签名者随机选取 $r_1,r_2 \in \mathbb{Z}_n^*$，然后计算

$$c = H(m \| y \| g \| h \| g^{r_1} h^{r_2}), \quad s_1 = r_1 - cx_1 \bmod n, \quad s_2 = r_2 - cx_2 \bmod n$$

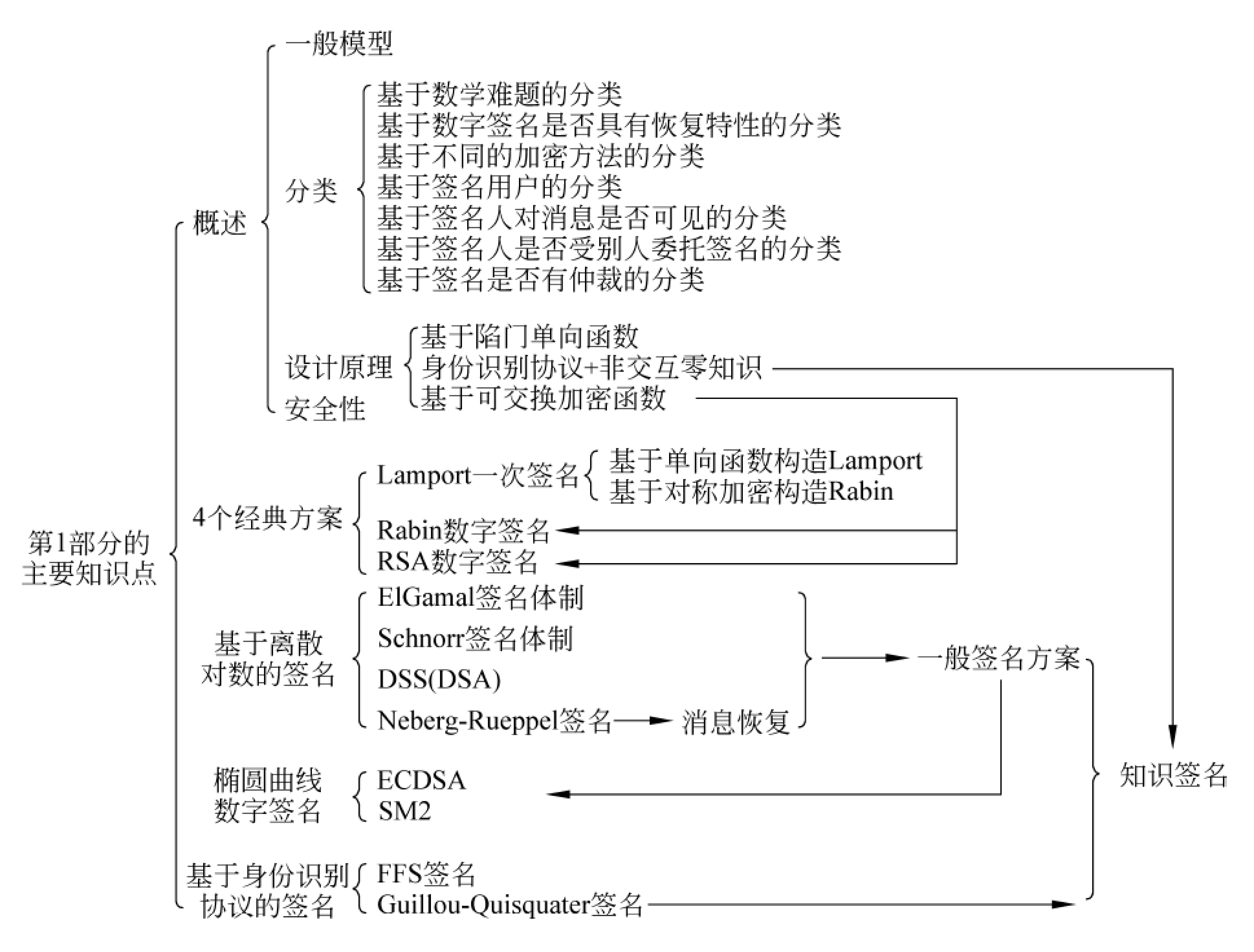
得到签名 (c,s_1,s_2)，由于任何不知道 (x_1,x_2) 的人都无法生成签名 (c,s_1,s_2)，所以这样一个签名可以证明 y 有着 $y=g^{s_1} h^{s_2}$ 这样的形式，并且签名者知道秘密值 (x_1,x_2)。

回顾前面介绍的基于离散对数的数字签名，定义 5.1 是 Schnorr 签名的一般化，定义 5.2 是 Okamoto 签名的一般化。

第 1部分

小结

本部分介绍了数字签名的基础知识。涵盖的主要知识点如下所示：



本部分的重点是基于离散对数的签名和椭圆曲线数字签名，难点是基于身份识别协议的签名，特别是知识签名。

扩展阅读建议

L. Lamport 签名以及其改进版本 Merkle 签名(又叫 Merkle 树):

L. Lamport. Constructing digital signatures from a one-way function. Technical Report SRICSL-98, SRI International Computer Science Laboratory, Oct. 1979.

Ralph Merkle. A certified digital signature. In Gilles Brassard, ed., Advances in Cryptology -- CRYPTO '89, vol. 435 of Lecture Notes in Computer Science, pp. 218-238, Spring Verlag, 1990.

Rabin 签名:

Michael O. Rabin. Digitalized signatures as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, Jan. 1979.

GMR 签名:

Shafi Goldwasser, Silvio Micali, Ronald Rivest. A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing, 1988, 17(2): 281-308.

ElGamal 系列签名:

T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans information Theory, 1985, 31(4): 469-472.

K. Nyberg, R. A. Rueppel. Message recovery for signature schemes based on the discrete logarithm problem. Designs, Codes and Cryptography, 1996, 7(1-2): 61-81.

Claus-Peter Schnorr. Efficient signature generation by Smart Cards. Journal of Cryptology 1991, 4(3): 161-174.

第 2 部分

高级数字签名

在现实生活中,数字签名的应用领域十分广泛,因此能适应某些特殊要求的数字签名技术也应运而生。如为了保护消息拥有者的隐私,要求签名人不能看见所签的消息,于是就有了盲签名的产生;签名人委托另一个人代表他签名,于是就有了代理签名的概念等。由于实际应用的需求,各种特殊的数字签名研究一直是数字签名领域非常活跃的部分,并产生了很多分支。本部分将分别介绍几种特殊数字签名。

盲签名(blind signature):一种让签名人不知道所签名文件内容的签名形式。它能使所签名文件的内容不被签名者获知,保护了个人的隐私。盲签名这一性质能结合到其他签名方式中,形成新的签名方式。如群盲签名、盲代理签名、代理盲签名、盲环签名等。

代理签名(proxy signature):将签名权委托给代理签名者,由他去代替自己行使签名。代理签名在电子商务中有着广泛的应用,也是与其他签名技术结合较多的一种签名形式。

多重签名(multiple signature):有多人参与对同一文件进行分别签名。可与其他方案结合派生出代理多重签名、多重盲签名等。

环签名(ring signature):一种与群签名有许多相似处的签名形式,它的签名者身份是不可追踪的,具有完全匿名性。

此外,还有失败-停止签名(fail-stop signature)、不可否认签名(undeniable signature)等分支。

第 6 章 盲 签 名

前面提到的签名都是自己给要签署的内容进行签名。下面看一个场景：如果你想让别人为你签署一份文件，但你又不想让别人看到这份文件，该如何实现？例如，你想让公证员签一个文件，公证员也不关心文件的内容，只是证明在某一个时刻公证过这个文件。

6.1 盲签名概念的提出与 Chaum 盲签名

盲签名(blind signature)的概念是 1982 年 D. Chaum 在 Cypto'82 上首次提出的一种特殊的数字签名。D. Chaum 形象地把盲签名比喻成在信封上签名，要签名的数据好比书信的内容，为了不使签名者看到数据，给信纸加一个具有复写能力的信封。这一过程称为盲化过程。经过盲化的文件，别人是不能读的。在盲化后的文件上签名，好比使用硬笔在信封上签名。虽然只是在信封上签名，但因信封具有复写能力，所以签名也会签到信封内的信纸上。

盲签名是公钥密码学中的一个重要的协议，可应用在电子货币中保护顾客的消费隐私以及在电子选举中对投票人的身份进行保密(匿名性)。例如，在电子货币的应用场景中，顾客 A 得到银行 B 对钱款 m 的盲签名后，自己算出银行的真正签名 $S_B(m)$；在支付时提交 m 和 $S_B(m)$；银行能验证 $S_B(m)$是否为 m 的合法签名，但不知道是谁的消费，从而保护了消费者的隐私。

盲签名方案中涉及两个主体：签名者和使用者。假设使用者有一条秘密消息需要签名者签名，但又不想让签名者知道该消息的内容。按照 Chaum 定义的盲签名协议，一个盲签名应该满足以下 3 个性质：

(1) 盲性(blindness)：签名者不知道他所签消息的内容。

(2) 不可追踪性(untraceability)：在签名公开后，签名者无法将他所签的消息与签名使用者联系起来。

(3) 无关性(unlinkability)：同一使用者的两条不同消息的签名不能建立起联系。

Chaum 盲签名的设计思想是：使用者先随机选一个或多个整数作为置盲因子，将他的秘密消息 m 盲化成消息 m'再发送给签名者。签名者用其私钥对消息 m'进行签名。使用者将签名者收到的签名脱盲，并对此签名用签名者的公钥进行验证。如验证正确，则使用者就得到 m 的一个有效签名，见图 6.1。

下面给出基于 RSA 公钥密码系统的 Chaum 盲签名。A 为签名者，B 为使用者。

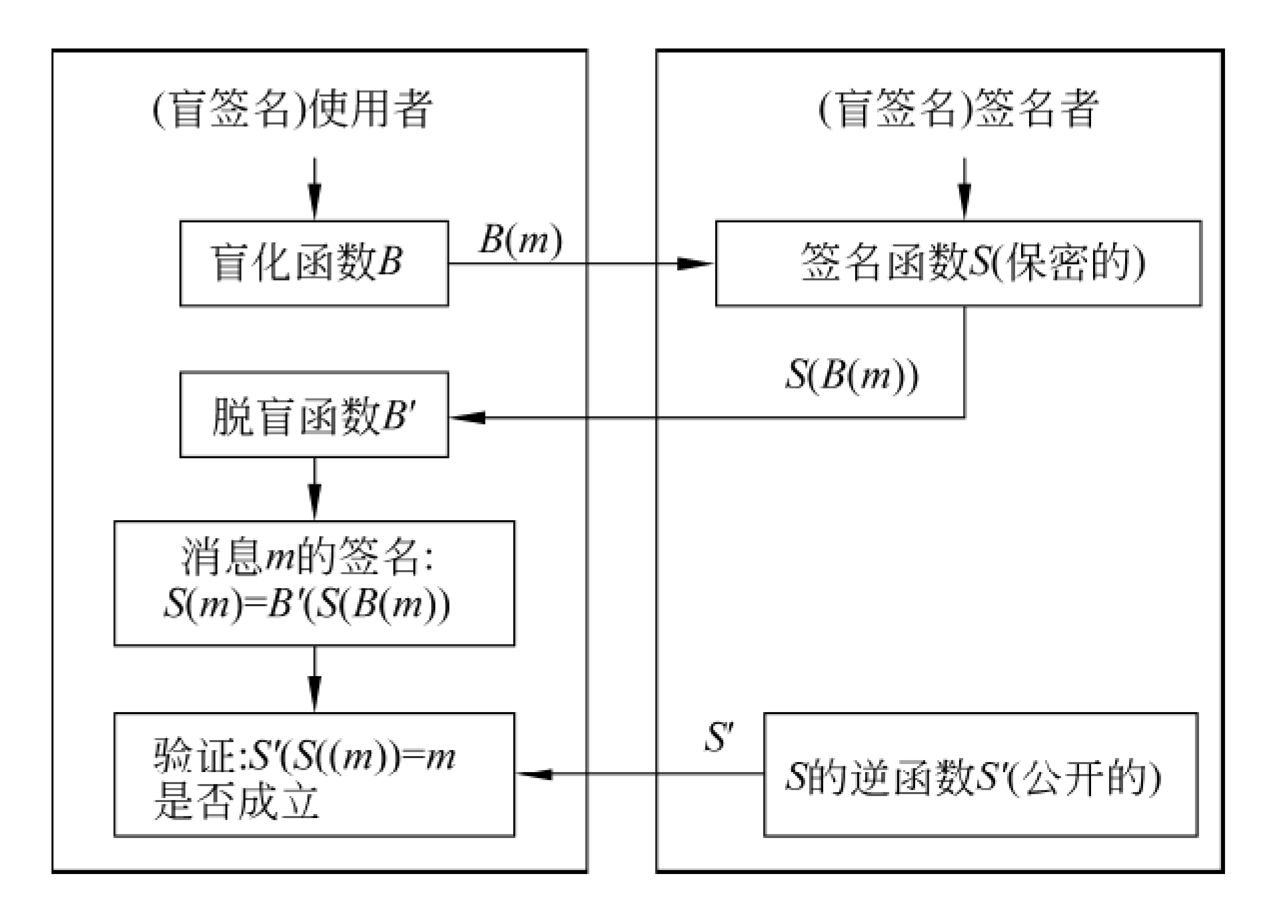


图 6.1 Chaum 盲签名方案的设计思路

(1) 密钥生成。选取素数 p,q,令 $n=p\times q$,随机选取 e:$1<e<\phi(n)$且 $\gcd(e,n)=1$,计算 d:$1<d<\phi(n)$,使得 $d\times e\equiv 1 \bmod \phi(n)$。将$(n,e)$作为 A 的公钥公开,$(p,q,d)$或者$(n,\phi(n),d)$作为 A 的私钥保密。

(2) 盲签名生成:

① 设 B 有消息 $m\in\mathbb{Z}_n$,B 随机选择 k:$1<k<n$ 及 $\gcd(k,n)=1$,置盲消息 m:$\overline{m}=mk^e \bmod n$,将 $\overline{m}$ 发送给 A。

② A 对消息 $\overline{m}$ 进行签名:$\bar{s}=\overline{m}^d \bmod n$,将 $\bar{s}$ 发送给 B。

③ B 进行脱盲计算:$s=k^{-1}\bar{s} \bmod n$,得到 A 对消息 m 的签名 s。

(3) 盲签名验证:

B 验证 $m=s^e \bmod n$。若成立,则接受签名,否则拒绝。

容易验证正确性。可见,Chaum 盲签名需要两个基本构件。

(1) 签名者 A 知道的盲化函数 B 以及脱盲函数 B'。B 与 B'必须满足 $B'(S(B(m)))=S(m)$。

(2) 签名者 B 的数字签名方案 S_B。

6.2 盲签名方案举例

6.2.1 基于 Schnorr 签名构造的盲签名

1992 年,Okamoto 在 Crypto'92 上基于 Schnorr 签名体制构造了第一种基于离散对手问题的盲签名方案①。使用者 B 让签名者 A 对 m 进行盲签名,过程如下:

(1) 参数生成:

签名者 A 随机选取两个足够大的素数 p,q,使得$\mathbb{Z}_p^*$ 和$\mathbb{Z}_q^*$ 上的离散对数问题都是困

---

① T. Okamoto. Provable Secure and Practical Identification Schemes and corresponding Digtial Signature Schemes. Crypto'92, 31-52.

难的。例如 $p\geqslant 2^{160}$，且 p,q 满足 $q|(p-1)$，$g\in\mathbb{Z}_p^*$ 满足 g 的阶为 q，即 $g^q=1 \bmod p$。秘密选取私钥 $x\in\mathbb{Z}_q^*$，计算：$y=g^x \bmod p$，(p,q,g,y) 作为公钥。另外，$h:\{0,1\}^*\to\mathbb{Z}_q^*$ 为一个安全散列函数。

(2) 盲签名生成：

① 签名者 A 随机选择 $\bar{k}\in\mathbb{Z}_q^*$，计算 $\bar{r}=g^{\bar{k}} \bmod p$，将 $\bar{r}$ 发送给消息拥有者 B。

② B 接收到 $\bar{r}$ 后，任取随机数 $\alpha,\beta\in\mathbb{Z}_q^*$，计算

$$r=\bar{r}g^{\alpha}y^{\beta} \bmod p$$
$$e=h(r\parallel m)$$
$$\bar{e}=e-\beta \bmod q$$

③ A 接到 e 后，计算签名 $\bar{s}=\bar{k}-\bar{e}x \bmod q$，将 $\bar{s}$ 发送给 B。

④ B 接收到 $\bar{s}$ 后，进行脱盲计算：$s=\bar{s}+\alpha \bmod q$，则 (e,s) 为消息 m 的盲签名。

(3) 盲签名验证：

计算 $t=g^s y^e \bmod p$，若 $h(t\parallel m)=e$ 成立，则 (e,s) 即为消息 m 的有效盲签名，否则是无效的签名。

正确性证明

$$\begin{aligned} t &= g^s y^e \bmod p = g^{\bar{s}+\alpha}y^{\bar{e}+\beta} \bmod p = g^{\bar{k}-\bar{e}x+\alpha}y^{\bar{e}+\beta} \bmod p \\ &= g^{\bar{k}-\bar{e}x+\alpha+\bar{e}x+\beta x} \bmod p = \bar{r}g^{\alpha}y^{\beta} \bmod p = r \end{aligned}$$

6.2.2 基于 Neberg-Rueppel 签名构造的盲签名

1994 年 Camenisch 等提出基于 Neberg-Rueppel 签名的盲签名方案。其特点是带有消息恢复功能的盲签名。使用者为 B，让签名者为 A 对消息 m 进行盲签名。执行以下过程：

(1) 密钥生成(与基于 Schnorr 签名的盲签名相同)：

签名者 A 随机选取两个足够大的素数 p,q，使得 $\mathbb{Z}_p^*$ 和 $\mathbb{Z}_q^*$ 上的离散对数问题都是困难的，例如 $p\geqslant 2^{160}$，且 p,q 满足 $q|(p-1)$，$g\in\mathbb{Z}_p^*$ 满足 g 的阶为 q，即 $g^q=1 \bmod p$。秘密选取私钥 $x\in\mathbb{Z}_q^*$，计算 $y=g^x \bmod p$，(p,q,g,y) 作为公钥。

(2) 盲签名生成：

① 签名者 A 随机选择 $\bar{k}\in\mathbb{Z}_q^*$，计算：$\bar{r}=g^{\bar{k}} \bmod p$，把 $\bar{r}$ 发送给 B。

② B 接收到 A 后，任取随机数 $\alpha,\beta\in\mathbb{Z}_q^*$，计算

$$r=mg^{\alpha\bar{r}\beta} \bmod p$$
$$\bar{m}=r\beta^{-1} \bmod q$$

发送 $\bar{m}$ 给 A。

③ A 接收到 $\bar{m}$ 后，利用自己的私钥 x 计算 $\bar{s}=\bar{k}+\bar{m}x \bmod q$，将 $\bar{s}$ 发送给 B。

④ B 接收到 $\bar{s}$ 后，计算 $s=\bar{s}\beta+\alpha \bmod q$，$(r,s)$ 即为消息 m 的盲签名。

(3) 消息恢复和盲签名验证：

接收者计算 $m=g^{-s}y^r r \bmod p$ 恢复消息。检查冗余度是否符合约定，符合约定则签名有效。

正确性证明

$$s=\bar{s}\beta+\alpha \bmod q=(\bar{k}+\bar{m}x)\beta+\alpha \bmod q$$
$$g^{-s}y^{r}r=g^{-(\bar{k}+\bar{m}x)\beta-\alpha}g^{xr}mg^{\alpha}\bar{r}^{\beta} \bmod p=g^{-\bar{m}x\beta}g^{xr}m \bmod p$$
$$=g^{x(-\bar{m}\beta+r)}m=m \bmod p$$

6.2.3 基于 ElGamal 签名构造的盲签名

1994 年 Camenisch 等提出另一个基于 ElGamal 签名构造的盲签名。使用者 B 让签名者 A 对消息 m 进行盲签名，执行以下过程：

(1) 密钥生成：(与基于 Schnorr 签名的盲签名相同)：

签名者 A 随机选取两个足够大的素数 p,q，使得 $\mathbb{Z}_p^*$ 和 $\mathbb{Z}_q^*$ 上的离散对数问题都是困难的。例如 $p\geqslant 2^{160}$，且 p,q 满足 $q|(p-1)$，$g\in\mathbb{Z}_p^*$ 满足 g 的阶为 q，即 $g^q=1 \bmod p$。秘密选取私钥 $x\in\mathbb{Z}_q^*$，计算 $y=g^x \bmod p$，(p,q,g,y) 作为公钥。

(2) 盲签名生成：

① 签名者 A 随机选择 $\bar{k}\in\mathbb{Z}_q^*$，计算 $\bar{r}=g^{\bar{k}} \bmod p$，把 r 发送给 B。

② 使用者 B 随机选择两个数 $\alpha,\beta\in\mathbb{Z}_q^*$，将消息置盲(盲化)：

$$r=g^{\alpha}\bar{r}^{\beta} \bmod p$$
$$\bar{m}=\beta m\bar{r}r^{-1} \bmod q$$

这里，把 $\bar{m}$ 发送给 A。

③ A 利用自己的私钥计算签名：$\bar{s}=(x\bar{r}+\bar{k}\bar{m}) \bmod q$，并把 $\bar{s}$ 发送给 B。

④ B 进行脱盲计算：$s=(\bar{s}r\bar{r}^{-1}+\alpha m) \bmod q$，这样 (r,s) 就是对消息 m 的一个盲签名。

(3) 盲签名验证：

签名验证过程为 $g^s\equiv y^r r^m \bmod p$。若成立，则接受签名，否则拒绝。验证正确性。因为

$$s\equiv(\bar{s}r\bar{r}^{-1}+\alpha m)\equiv(x\bar{r}+\bar{k}\bar{m})r\bar{r}^{-1}+\alpha m$$
$$\equiv xr+\bar{k}\bar{m}r\bar{r}^{-1}+\alpha m\equiv xr+\alpha m+\bar{k}\beta m \bmod q$$

故

$$g^s\equiv g^{xr+\alpha m+\bar{k}\beta m \bmod q}\equiv g^{xr}g^{(\alpha+\bar{k}\beta)m}\equiv y^r(g^{\alpha}(g^{\bar{k}})^{\beta})^m$$
$$\equiv y^r(g^{\alpha}\bar{r}^{\beta})^m\equiv y^r r^m \bmod p$$

思考 6.1 一个很自然的问题就是该方案是如何设计的(设计原理)。

答案见 6.2.4 节。

6.2.4 ElGamal 型盲签名方案的一般构造方法*

参数产生方法类似，重点讨论签名方案和脱盲方程的设计方法。

设签名者 A 对盲消息的签名方程的模型为 $a'k=b'+c'x \bmod q$，其中 (a',b',c') 为 $(\pm\bar{r},\pm\bar{m},\pm\bar{s})$ 的某一置换或线性组合。

使用者 B 脱盲后的签名变量 s,r 也必须满足相应的签名方程 $ak=b+cx \bmod q$，其中 (a,b,c) 为 $(\pm r,\pm m,\pm s)$ 的线性组合。

就6.2.3节中的例子(基于ElGamal签名构造的盲签名)而言,签名者A的盲签名方程为 $\bar{s}=(x\bar{r}+\bar{k}\bar{m}) \bmod q$,如果要求脱盲后的签名方程为 $s=rx+km \bmod q$,又因为 $r=g^{\alpha}\bar{r}^{\beta} \bmod p$,即 $r=g^{\alpha}\bar{r}^{\beta} \bmod p=g^{\alpha+\bar{k}\beta} \bmod p$,即有 $k=\alpha+\bar{k}\beta \bmod q$(因为在非盲签名中有 $r=g^k \bmod q$),于是得到方程组:

$$\bar{s}=(x\bar{r}+\bar{k}\bar{m}) \bmod q \tag{1}$$
$$s=rx+km \bmod q \tag{2}$$
$$k=\alpha+\bar{k}\beta \bmod q \tag{3}$$

解此方程组,由式(2)得 $x=r^{-1}(s-km)$,代入到式(1)得(为简洁,省去写出 mod q)

$$\bar{r}r^{-1}(s-km)+\bar{k}\bar{m}-\bar{s}=0$$

将式(3)代入上式:

$$\bar{r}r^{-1}(s-(\alpha+\bar{k}\beta)m)+\bar{k}\bar{m}-\bar{s}=0$$

该式为关于 $\bar{k}$ 的恒等式,所以有

$$\bar{r}r^{-1}(s-\alpha m)-\bar{s}=0$$
$$-\bar{r}r^{-1}\beta m+\bar{m}=0$$

于是

$$s=\bar{s}r\bar{r}^{-1}+\alpha m \bmod q$$
$$\bar{m}=\beta m\bar{r}r^{-1} \bmod q$$

这就是方案中所使用的脱盲方程和消息盲化方法。

以此类推,可以设计18种广义ElGamal盲签名,有兴趣的读者可参阅相关文献。

6.3　盲签名的应用

盲签名广泛应用在电子选举和电子货币中。这里以电子选举为例说明其如何应用。

设A是选举管理中心,B是选民,v 是选票,ID_B 是选民B的身份信息。B不想让A知道其选票的内容。但是,任何一张选票产生后,必须先经过管理中心对投票的选民身份进行确认,然后对选票进行签名后才能生效。因此,选民B填好选票 v 后,先对选票 v 用盲变换Blind进行盲化,得到Blind(v),然后对Blind(v)签名,得到 $s=\text{Sig}_B(\text{Blind}(v))$,再将($ID_B$,Blind($v$),$s$)发送给A。

选举管理中心A收到(ID_B,Blind(v),s)后,执行如下步骤:

(1) 检查B有无权利参加选举。若B无权参加选举,则将B的选票Blind(v)作废,不予签名。否则进行下一步。

(2) 检查B是否已经参加过投票,即检查Blind(v)是否为重复投票。若已投过票,则将B的选票作废,不予签名。否则,进行下一步。

(3) 检测 s 是否是选票Blind(v)的有效签名。若不是,则将B的选票Blind(v)作废,不予签名。否则,对B的选票签名,得到 $s'=\text{Sig}_A(\text{Blind}(v))$,并把 s' 发送给选民B。

最后,选举管理中心A宣布对选票签名的总人数,并公布(ID_B,Blind(v),s)的列表。

选民B获得 s' 后,验证A的签名是否有效。若无效,要重新向A申请对自己的选票进行签名。如果A的签名有效,则B将从 s' 中获得A对 v 的签名 s'',然后匿名地将(s'',v)

发送给计票站。

例 6.1 一个基于 RSA 的盲签名的简单电子投票系统。

假设 A 和 B 采用的都是基于 RSA 的签名算法，A 的公钥为 e_A，私钥为 d_A，模数为 n_A；B 的公钥为 e_B，私钥为 d_B，模数为 n_B；设 h 是一个安全散列算法。B 可向 A 提出申请，A 先检查 B 的选民身份是否合格，如合格，则发送给 B 一个随机数 $0<r_B<\min(n_A,n_B)$。B 收到参数 r_B 后，再选择一个随机数 $r<\min(n_A,n_B)$，如下产生 Blind(v)和 s：

$$\text{Blind}(v)=h(v)r^{e_A} \bmod n_B, s=(\text{Blind}(v))^{d_B}r_B \bmod n_B$$

A 收到 B 的数据(ID_B，Blind(v)，s)后，验证下式是否成立：

$$\text{Blind}(v)=(s/r_B)^{e_B} \bmod n_A$$

如果成立，则计算签名

$$s'=\text{Blind}(v)^{d_A} \bmod n_A$$

B 收到 s'后，容易验证签名是否有效。然后计算

$$s''=s'/r=h(v)^{d_A} \bmod n_A$$

这是因为

$$s'=(h(v)r^{e_A})^{d_A}=h(v)^{d_A}r \bmod n_A$$

最后，B 将(s''，v)匿名发送给计票站。

这个过程中，A 虽然对 $h(v)$做了签名，但 A 并不知道 v 的内容，也不知道签名 s''。方案中加上随机数 r，是为了即使在投票结束后，A 也无法通过比较 v 和 Blind(v)来获知 v 的投票者 B 的身份。而增加随机数 r_B，是为了防止重放 B 过去的投票，致使 B 无法参与投票。

第7章 代理签名

先看一个场景，公司的经理外出度假期间，让他的秘书代理公司的业务，包括以自己的名义在一些文件上签名。通常是经理将自己的名章交给秘书，让秘书代表自己盖章，即将自己的签名权委托给代理人。

数字签名权利的委托(delegation of the digital signing power)是数字化的信息社会必然遇到的一种现象，但是，将自己的签名权利委托他人的时候，必须考虑以下几个问题：

(1) 安全性(security)。一般来说，一个人将数字签名权力委托给代理人的时候，希望代理人只能代表他在特定的时间对特定的文件生成数字签名，而不希望代理人"滥用"他的数字签名权力，且不希望非法的攻击者能因此伪造出有效的数字签名。

(2) 实用性(practicability)。委托数字签名权力的方法方便、有效，容易实现。

(3) 效率(efficiency)。委托权力的方法具有较高的速度和较小的计算复杂性、通信复杂性等。

7.1 代理签名的基本概念和分类

代理签名(proxy signature)，又称为委托签名，是1996年由Mambo、Usuda和Okamoto在ACM CCS96会议上首次提出的[①]。代理签名是指在一个签名方案中，原始签名人(original signer)把他的签名权授予代理签名人(proxy signer)，然后代理签名人代表原始签名人生成有效的签名。

一个代理签名体制可由以下几个部分组成：

(1) 参数产生。选定签名体制的参数、用户的密钥等。

(2) 数字签名权利的委托。原始签名人将自己的签名权利委托给代理签名人。

(3) 代理签名的生成。代理签名人代表原始签名人生成数字签名。

(4) 代理签名的验证过程。验证人验证代理签名的有效性。

以上是一个直观的定义，下面给出一个形式化的定义。这也是定义签名方案的第一步。

定义7.1 设A,B是一个数字签名方案(M,S,SK,PK,GenKey,Sign,Vrfy)的两个用户，其中M是消息集合，S是签名集合，SK是私钥集合，PK是公钥集合，GenKey表示

---

① M. Mambo, K. Usuda, E. Okamoto. Proxy Signature for Delegating Signing Operation. Proc. of ACM CCS96, 48-57.

密钥生产算法集合，Sign 表示签名生成算法的集合，Vrfy 是签名验证算法集合，它们的私钥、公钥分别是(x_A, y_A)，$(x_B, y_B) \in SK \times PK$。如果以下条件成立：

(1) A 利用他的私钥 x_A 计算出一个数 δ，然后将 δ 秘密交给 B；任何人(包括 B)在试图求出 x_A 时，δ 不会对求解有任何帮助。

(2) 代理签名者 B 可以利用 δ 和 x_B 生成一个新的签名密钥 $\delta_{A\to B}$。

(3) 存在一个公开的验证算法 $\mathrm{Vrfy}_{A\to B}: PK \times S \times M \to \{\mathrm{True}, \mathrm{False}\}$，使得对任何 $s \in S$ 和 $m \in M$，都有 $\mathrm{Vrfy}_{A\to B}(y_A, s, m) = \mathrm{True} \Leftrightarrow s = \mathrm{Sign}(\delta_{A\to B}, m)$。

(4) 任何人在试图求解 x_A，x_B，δ 和 $\delta_{A\to B}$时，任何数字签名 $\mathrm{Sign}(\delta_{A\to B}, m)$都不会对求解有任何帮助。

则称用户 A 将他的签名权力委托给用户 B，且称 A 为原始签名人，B 为 A 的代理签名人，δ 为委托密钥(delegating key)，$\delta_{A\to B}$为代理签名密钥(proxy signing key)，以代理签名密钥对消息 m 生成的签名 $\mathrm{Sign}(\delta_{A\to B}, m)$为 A 的代理签名，能够生成代理签名的数字签名体制称为代理签名体制。

代理签名体制应满足如下基本性质：

(1) 不可伪造性(unforgeability)。除了原始签名人外，任何人(包括代理签名人)都不能生成原始签名人的普通数字签名。这个性质是数字签名体制的基本要求，保证了原始签名人的基本安全要求。

(2) 代理签名的不可伪造性。除了代理签名人外，任何人(包括原始签名人)都不能生成有效的代理签名。如果原始签名人委托了多个代理签名人，那么任何代理签名人都不能伪造其他代理签名人的代理签名。这一性质保证了代理签名人的基本安全需求。

(3) 代理签名的可区分性(distinguishability)。任何一个代理签名都与原始签名人的普通数字签名有明显的区别，不同的代理签名人生成的代理签名之间也有明显的区别。这个性质和性质(1)、(2)结合起来可防止签名人之间的互相抵赖。

(4) 不可抵赖性(undeniability)。任何签名人(不论是原始签名人还是代理签名人)在生成一个数字签名后，不能再对它加以否认。这个性质可由性质(1)～(3)推导出来。

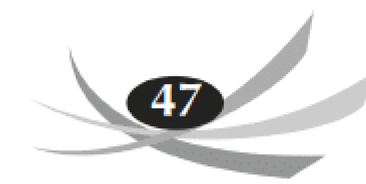
(5) 身份可识别性(identifiability)。原始签名人可以根据一个有效的代理签名确定相应的代理签名人的身份。利用这个性质，原始签名人可以对代理签名人进行监督，使代理签名人不能在不被发现的情况下滥用他的代理签名权力。

Mambo 等将代理签名分为 3 类：完全代理签名(full delegation)、部分代理签名(partial delegation)、具有证书的代理签名(delegation by warrant)。

(1) 完全代理签名即原始签名人直接将自己的私钥通过安全信道发送给代理签名人，这是一种平凡的代理签名。由于代理签名人产生的签名与原始签名者产生的签名是不可区分的，故不能制止签名滥用，且不具备可识别性。

(2) 部分代理签名中，代理签名的密钥是由原始签名人的密钥计算出来的，但由代理密钥计算不出原始签名的密钥。签名时用到原始签名人的公钥。

(3) 具有证书的代理签名方案中使用了一个称为委任状的文件来实现签名权的委托。代理签名人在签名时，用自己的签名密钥进行签名。一个有效的代理签名由代理签名人生成的签名和原始委任状组成。



部分代理签名和具有证书的代理签名比完全代理签名安全，而部分代理签名比具有证书的代理签名灵活、方便，故本节主要讨论部分代理签名。

部分代理签名可分为两种类型：

（1）不保护代理的代理签名（proxy-unprotected proxy signature）。指定的代理签名者能够代表原始签名者产生有效代理签名，没有指定为代理签名者的第三方不能产生有效代理签名，但是原始签名者可产生有效代理签名，这时代理签名密钥就是 δ。

（2）保护代理的代理签名（proxy-protected proxy signature）。指定的代理签名者能代表原始签名者产生有效代理签名。第三方都不能产生有效代理签名，而且原始签名者也不能产生有效代理签名。这是因为代理签名密钥由 δ 和代理签名人的私钥 x_B 两部分组成。

7.2　代理签名举例

7.2.1　MUO不保护代理的代理签名

下面介绍 M. Mambo、K. Usuda 和 E. Okamoto 提出的 MUO 方案。

假设$(M,S,\mathrm{SK},\mathrm{PK},\mathrm{GenKey},\mathrm{Sign},\mathrm{Vrfy})$是一个基于离散对数问题的数字签名体制，$p$ 是一个大素数，q 为 $p-1$ 或 $p-1$ 的大素数因子；$g\in\mathbb{Z}_p^*$，且 $g^q=1 \bmod p$。用户 A、B 的私钥和公钥分别是(x_A,y_A)，(x_B,y_B)，满足 $y_A=g^{x_A} \bmod p$，$y_B=g^{x_B} \bmod p$。

（1）委托过程。

① A 随机选取一个数 $k\in\mathbb{Z}_p^*$，计算 $K=g^k \bmod p$。

② A 计算 $\delta=x_A+kK \bmod q$，将(δ,K)秘密发送给 B。

③ B 验证等式 $g^\delta=y_AK^K \bmod p$ 是否成立，如不成立，则要求 A 重新执行步骤①或终止。

（2）代理签名的生成。

对消息 m，B 使用 δ 生成普通的数字签名 $s=\mathrm{Sign}_\delta(m)$，然后将$(s,K)$作为代表 A 对消息 m 生成的数字签名，即代理签名。

（3）代理签名的验证。接收方收到了消息 m 和代理签名(s,K)，按如下步骤来验证代理签名的有效性：

① 计算 $v=y_AK^K \bmod p$。

② 验证 $\mathrm{Vrfy}(y_A,(s,K),m)=\mathrm{True}\Leftrightarrow\mathrm{Vrfy}(v,s,m)=\mathrm{True}$。

MUO 方案具有以下几个性质：

（1）基本的不可伪造性。B 难以根据他所得到的(δ,K)计算出 x_A，从而不能伪造 A 的普通数字签名。同时也说明任何其他攻击者难以伪造 A 的普通数字签名。

（2）代理签名的不可伪造性。由于 A 和 B 都知道(δ,K)，所以 A 和 B 都能生成代理签名（也说明这是不保护代理的代理签名）。但是除了 A 和 B 以外，其他任何人都难以伪造一个有效的代理签名。

（3）代理签名的可区分性。代理签名(s,K)由两部分组成，一部分是普通数字签名

s，另一部分是某个数 K。由于代理签名比普通签名多出一部分（即 K），容易将代理签名与普通的数字签名区分开来。同时，不同的代理签名人的代理签名也可区分，假如除 B 外还有代理签名人 C，A 在委托过程中发送给 C 的消息是 (δ', K')，其中 $K'=g^{k'} \bmod p$，$k\neq k'$，于是 $K\neq K'$，C 生成的代理签名为 (s', K')，于是可将 B 和 C 的代理签名区分开来。

（4）不可抵赖性。由于任何人都不能伪造 A 的普通数字签名，所以 A 不能否认其有效的数字签名。由于除了 A 和 B 外，任何人都不能伪造 B 的代理签名，所以 A 和 B 不能否认一个有效的代理签名，即一个有效的代理签名必然是 A 和 B 两者中的一个生成的。但是，A 和 B 之间可以相互抵赖，即声称代理签名是对方而不是自己生成的。

（5）身份可识别性。在这个代理签名方案中，如果 A 在向 B 发送 (δ, K) 时，将 K 和 B 的身份保存在一起，那么当 A 看到一个有效的代理签名 (s, K) 时，就可以通过 K 识别 B 的身份。

上面给出的方案的步骤(2)中没有给出具体的签名方案。这里给出一个完整的步骤(2)，并给出对应的步骤(3)。参数设置和原步骤(1)不变。

（1）委托过程。

步骤同前。

（2）代理签名的生成。对消息 m，B 执行如下操作：

① 选择随机数 $r\in\mathbb{Z}_q^*$，计算 $R=g^r \bmod p$。

② 计算 $s=r^{-1}(m-\delta R) \bmod (p-1)$。

代理签名为 (R, s, K)。

（3）代理签名的验证。接收方收到了消息 m 和代理签名 (R, s, K)，按如下步骤来验证代理签名的有效性：

① 计算 $v=y_A K^K \bmod p$。

② 验证 $g^m=R^s v^R \bmod p$ 是否成立，如果成立，则代理签名正确。

正确性证明留作练习。

通过完整的方案可以体会到，代理签名人用委托密钥 δ 签名，验证签名时用到原始签名人的公钥，$v=y_A K^K \bmod p$ 其实可视为"委托密钥对应的公钥"。委托密钥和原始签名人的公钥通过一个随机的"承诺"K 联系起来。

7.2.2 MUO 保护代理的代理签名

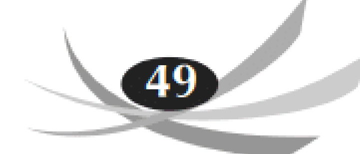
7.2.1 节介绍了不保护代理的代理签名，即 A 和 B 都能生成代理签名。保护代理的代理签名的设计目标是使得只有代理签名人才可以生成代理签名，方法是：在步骤(2)中不用委托密钥 δ 签名，而是用代理签名密钥 $\delta_{A\to B}$ 签名，对消息 m 生成的签名 $\mathrm{Sign}(\delta_{A\to B}, m)$ 为 A 的代理签名，而不是将 $\mathrm{Sign}(\delta, m)$ 作为 A 的代理签名。

注意：代理签名密钥 $(\delta_{A\to B})$ 由委托密钥 δ 和代理签名人的私钥生成，故只有代理签名人可以生成代理签名。

于是，得到保护代理的 MUO 代理签名方案。参数生成过程同 7.2.1 节。

（1）委托过程。

① A 随机选取一个数 $k\in\mathbb{Z}_p^*$，计算 $K=g^k \bmod p$。



② A 计算 $\delta = x_A + kK \bmod q$,将$(\delta, K)$秘密发送给 B。

③ B 验证等式 $g^{\delta} = y_A K^K \bmod p$ 是否成立,如不成立,则要求 A 重新执行步骤①或终止。

④ B 计算 $\bar{\delta} = \delta + x_B y_B \bmod q$。

(2) 代理签名的生成。对消息 m,B 使用 $\bar{\delta}$ 生成普通的数字签名 $s = \mathrm{Sign}_{\bar{\delta}}(m)$,然后将$(s, K)$作为代表 A 对消息 m 生成的数字签名,即代理签名。

(3) 代理签名的验证。

接收方收到了消息 m 和代理签名(s, K),按如下步骤来验证代理签名的有效性:

① 计算 $v = y_A K^K y_B^{y_B} \bmod p$。

② 验证 $\mathrm{Vrfy}(y_A, y_B, (s, K), m) = \mathrm{True} \Leftrightarrow \mathrm{Vrfy}(v, s, m) = \mathrm{True}$。

第 8 章

多重数字签名

8.1 多重数字签名的基本概念

首先看两个场景：①一份合同的签订，需要有甲乙双方的签字，有时甚至是多方（加上公证方）的签字。②一份员工入职批准文件，可能需要人事部、财务部、开发部等多个部门签字才能有效。由此提出一个要求——如何实现多个用户对同一个消息的数字签名。

1983 年 Boyd 提出了多重数字签名（digital multisignature）的概念，即可实现多个用户对同一消息进行数字签名。根据签名顺序划分，可将多重数字签名方案分为广播多重数字签名（broadcasting multisignature）和顺序多重数字签名（sequential multisignature）。广播多重数字签名是指消息发送者将消息发送给每一位签名者进行签名，然后签名者将签名消息发送给签名收集者，由收集者对签名消息进行整理而形成签名。顺序多重数字签名是指每一位签名者按照一定的顺序进行签名。

通常多重数字签名方案的参与者有消息发送者（issuer），消息签名者（signer），签名验证者（verifier）和签名收集者（collector）。多重签名的过程一般由以下几个部分组成：

（1）参数的生成。选定签名体制的参数、用户的密钥等。

（2）单个签名人签名的生成。签名人按照协议要求对消息进行签名。

（3）多重签名的产生。按照协议要求，生成总的签名。

（4）签名的验证。验证人验证签名的有效性。

图 8.1 是两种多重数字签名方案的示意图。

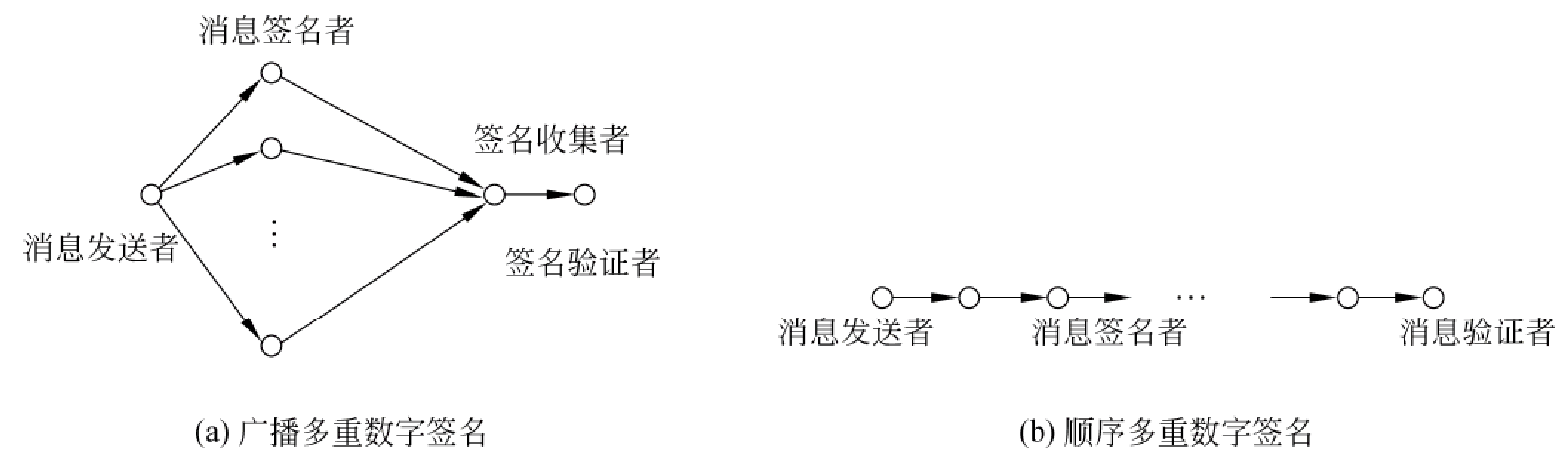


图 8.1 两种多重数字签名方案的示意图

多重数字签名的性质如下：

(1) 不可伪造性。这是数字签名的一个基本要求，这里指除签名组成员共同完成签名外，任何人不可能伪造多重签名，包括签名组成员中的一部分也不可能合谋产生有效签名。

(2) 不诚实签名者的可识别性。如果签名组成员中有不诚实者试图伪造签名，则在签名过程中或验证过程中就能被发现。如果签名组中有不诚实者，那么签名不可能完成。

(3) 不可否认性。签名一旦形成，签名组全体成员不能否认其签名。

(4) 可验证性。消息接收人或签名验证人能根据协议要求检验签名的真伪。

8.2　多重数字签名举例

8.2.1　ElGamal 型广播多重数字签名

广播多重数字签名中，签名组成员没有签名的先后顺序，但要求必须由一个签名收集者最终完成整个签名。下面介绍 ElGamal 型多重数字签名方案。

方案的参与者有消息发送者 U_I、若干签名者 $U_i(i=1,2,\cdots,n)$、签名收集者 U_C 和签名验证者 U_v。

(1) 参数产生。每一位签名者 $U_i(i=1,2,\cdots,n)$任意选取 $x_i\in[1,p-1]$作为 U_i 的私钥，并计算 $y_i=g^{x_i} \bmod p$，发送给其他每一位签名者 $U_j(j\neq i)$。

(2) 签名生成。

① U_I 将 m 发送给每一位签名者 $U_i(i=1,2,\cdots,n)$，U_i 随机选取 $k_i\in[1,p-1]$，计算 $r_i=g^{k_i} \bmod p$，发送给其他每一位签名者 $U_j(j\neq i)$。

② U_j 收到 $r_i(i=1,2,\cdots,n)$后计算

$$R=\prod_{i=1}^{n} r_i^{r_i} \bmod p$$

然后计算 $s_i=(R+h(m))x_i-r_ik_i \bmod (p-1)$，将签名$(m,(s_i,r_i))$发送给 U_C。U_C 收到$(m,(s_i,r_i))$，$(i=1,2,\cdots,n)$后，首先计算 R，然后验证方程 $y_i^{R+h(m)}=r_i^{r_i}g^{s_i} \bmod p$ 是否成立。若成立，说明 U_i 的签名是正确的，否则，不能接受签名，要求重新签名或放弃本次多重签名。

U_C 计算 $s=s_1+s_2+\cdots+s_n$，则多重签名是$(m,(s,R))$。

(3) 签名验证。当 U_v 得到消息及其签名$(m,(s,R))$后，验证等式

$$Rg^s=(y_1y_2\cdots y_n)^{R+h(m)}$$

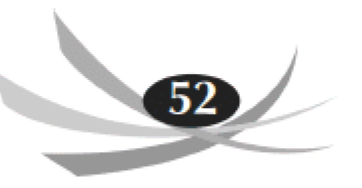
如果成立，则认为签名有效，否则签名无效。

容易得到正确性证明。

$g^s=g^{s_1}g^{s_2}\cdots g^{s_n} \bmod p$，有

$$g^s=\prod_{i=1}^{n}(r_i^{-r_i}y_i^{R+h(m)}) \bmod p$$

$$\prod_{i=1}^{n}(r_i^{r_i})g^s=\prod_{i=1}^{n}(y_i^{R+h(m)}) \bmod p$$



$$Rg^{s} = \prod_{i=1}^{n}(y_i^{R+h(m)}) \bmod p$$

8.2.2 ElGamal 型顺序多重数字签名

顺序多重签名要求所有的签名者按照一定的次序进行签名，而不是所有签名者分别对某一个消息同时签名。与普通数字签名相比，顺序多重数字签名还有以下特征：

(1) 签名的长度与签名人的数目无关。

(2) 不必知道每个签名者的公钥，而使用组公钥就可以验证签名。

(3) 签名者必须按照特定的次序依次对消息进行签名，否则无法获得有效的签名。

(4) 没有所有的签名者联合操作，要获得有效的有序签名，在计算上是不可行的。

顺序多重数字签名的参数产生同 8.2.1 节。消息发送者预先设计一个签名顺序 U_1，U_2，…，U_n。

下面介绍签名过程。U_I 把消息 m 发送给第一个签名者 U_1，设 $s_0=0$，那么每个签名者 $U_i(i\geqslant 2)$收到上一个签名者 U_{i-1}的签名(m, s_{i-1})后，做如下操作：

(1) 验证等式

$$g^{s_{i-1}}\prod_{j=1}^{i-1} r_j^{r_j} \equiv \prod_{j=1}^{i-1} y_j^{h(m)} \bmod p$$

若成立，则继续，否则拒绝对消息签名。

(2) 验证成立后，U_i 随机选择 $k_i \in [1, p-1]$，计算

$$r_i \equiv g^{k_i} \bmod p$$

$$s_i \equiv s_{i-1} + h(m)x_i - r_i k_i \bmod (p-1)$$

然后将(m, s_i)发送到下一个签名者 U_{i+1}，将 r_i 分别发送到 U_i 以后的签名者。

文件签名者 U_i 完成签名后生成消息 m 的顺序多重签名$(m, s_t, r_1, r_2, \cdots, r_i)$。

(3) 签名验证。消息 m 的顺序多重签名为$(m, s_t, r_1, r_2, \cdots, r_i)$。签名验证者 U_v 验证等式

$$g^{s_t}\prod_{j=1}^{t} r_j^{r_j} \equiv \prod_{j=1}^{t} y_j^{h(m)} \bmod p$$

如果等式成立，则 U_v 认为 U_1，U_2，…，U_t 对消息 m 进行的顺序多重签名有效，否则无效。正确性证明容易得到：

由于

$$s_i \equiv s_{i-1} + h(m)x_i - r_i k_i \bmod (p-1)$$

于是

$$s_i \equiv \sum_{j=1}^{i}(h(m)x_j - r_j k_j) \bmod (p-1)$$

即

$$\sum_{j=1}^{i} r_j k_j + s_i \equiv \sum_{j=1}^{i} h(m)x_j \bmod (p-1)$$

于是，对任意 i，有

$$g^{\sum_{j=1}^{i} r_j k_j + s_i \bmod q} \equiv g^{\sum_{j=1}^{i} h(m) x_j \bmod q} \bmod p$$

因此

$$g^{s_i} \prod_{j=1}^{i} r_j^{r_j} \equiv \prod_{j=1}^{i} y_i^{h(m)} \bmod p$$

第 9 章

其他高级签名*

9.1 环　签　名

先看一个场景：假设 Alice 是某个国家的内阁成员，她知道一条关于首相的丑闻，并想将这个丑闻泄露给报刊记者。Alice 不能让一个平民百姓去告诉记者，因为这样的检举别人不会相信。但是 Alice 又不能用普通的数字签名，因为这样会暴露自己的身份。Alice 采取的办法是选择所有的内阁成员构成一个“环”，使用代表这个“环”的签名发送给记者，记者验证签名，可以确信消息是由内阁中的某个成员泄露的，从而具有可信性；但同时记者无从获知检举人的身份，猜中是 Alice 检举的机会只是 $1/n$，从而实现 Alice 的匿名检举的目的。

9.1.1 环签名的基本概念

2001 年的 Asiacrypt 会议上，R. L. Rivest 和 A. Shamir 等人以如何泄露秘密为背景提出环签名(ring signature)的概念[①]。其基本思想是：签名人可以随意将自己包括在一个群体(构成一个环)内，在签名时把所有群体成员的信息(公钥)作为签名的一部分。任何人可以验证签名是群体中的一个成员所做的，但无法推断出到底是哪个成员所做的。群体中的任何一个成员是真正的签名者的概率都相等。因此，环签名可以实现对签名者的无条件匿名。正是由于这一点，它非常适合一个群体的成员以群体的名义对外泄露秘密，而不至于事后被查出究竟是谁。当然，除了泄露秘密外，环签名可用于群体成员的匿名认证等方面。

定义 9.1　环签名的基本概念如下：

给定一个环 $U=\{U_1,U_2,\cdots,U_n\}$，环中每个用户的公钥—私钥对为(pk_i,sk_i)，$i=1,2,\cdots,n$。不失一般性，假设 $U_k(1\leqslant k\leqslant n)$是签名人。除密钥生成算法外，一个环签名体制还包含环签名产生算法 ring-sign 和环签名验证算法 ring-verify：

(1) ring-sign，环签名产生算法。输入是待签名的消息 m、环中所有成员的公钥 $pk_i(1\leqslant i\leqslant n)$和真正签名人的私钥 sk_k；输出是 U_k 对消息 m 的环签名 s，记作 $s\leftarrow \text{ring-sign}(m,pk_1,pk_2,\cdots,pk_n,sk_k)$。

(2) ring-verify，环签名验证算法。输入是待验证的消息签名对(m,s)和环中所有成员的公钥；输出为签名验证正确或错误，即$\{\text{True},\text{False}\}\leftarrow \text{ring-verify}(m,s,pk_1,pk_2,\cdots,pk_n)$。

---

① R. Rivest, A. Shamir, Y. Tauman. How to Leak a Secret. Asiacrypt01, 2001: 552-565.

环签名的安全性一般要求如下：

(1) 正确性(consistency)。环中的任一成员执行环签名产生算法后输出的签名都能通过该体制中的签名验证算法。

(2) 匿名性(anonymity)。给定一个环签名，则任意验证者不会以大于 $1/n$ 的概率识别产生该签名的真正签名人，其中 n 为环成员个数。

(3) 不可伪造性(unforgeablity)。任意不在环上的用户不能有效地产生一个消息签名对 (m,s)，使得 ring-verify$(m,s,pk_1,pk_2,\cdots,pk_n)=$True。

9.1.2 第一个环签名方案

在此介绍 R. Rivest 和 A. Shamir 等人提出的第一个环签名方案。

1. 基本假设与参数

设 r 个群体成员为 $A_1,A_2,\cdots,A_r$，实际签名人为 A_s，$1\leqslant s\leqslant r$。

A_i 的公钥为 P_i，消息是长度为 b 的比特串，每一个公钥 P_i 对应一个单向陷门置换 $g_i:\{0,1\}^b\rightarrow\{0,1\}^b$，只有在知道 P_i 对应的私钥的情况下才可以求出 g_i 的逆。其实，g_i 就是一个公钥加密算法。

E 是一个密钥长度为 l 的公开的对称加密算法，每一个密钥对应的加密函数是一个置换 $E_k:\{0,1\}^b\rightarrow\{0,1\}^b$。

$h:\{0,1\}^*\rightarrow\{0,1\}^l$ 是公开的抗碰撞散列函数。

$\{C_{k,v}(y_1,y_2,\cdots,y_r):k\in\{0,1\}^l,v\in\{0,1\}^b\}$ 是一个合成函数，其中 k 是密钥，v 是初始值。对确定的 k 和 v，$C_{k,v}(y_1,y_2,\cdots,y_r):\{0,1\}^b\times\{0,1\}^b\times\cdots\times\{0,1\}^b\rightarrow\{0,1\}^b$ 满足下述 3 条性质：

(1) 对每一个 s，$1\leqslant s\leqslant r$，在其他的输入 $y_i(i\neq s)$ 被固定时，$C_{k,v}(y_1,y_2,\cdots,y_r)$ 是 $\{0,1\}^b$ 上的置换。

(2) 对每一个 s，$1\leqslant s\leqslant r$，给定一个 b 比特值 z 和除了 y_s 外的其他 y_i 值，可有效地计算出 y_s 的一个值，使得 $C_{k,v}(y_1,y_2,\cdots,y_r)=z$，即环签名是容易计算的。

(3) 给定 k,v,z 的值，在不知道任何单向陷门函数 $g_1,g_2,\cdots,g_r$ 的逆的情况下，要求出方程 $C_{k,v}(g_1(x_1),g_2(x_2),\cdots,g_r(x_r))=z$ 的解 $x_1,x_2,\cdots,x_r$ 是不可行的。

一个具体的 $C_{k,v}(y_1,y_2,\cdots,y_r)$ 可用对称加密函数 E_k 构造：

$$C_{k,v}(y_1,y_2,\cdots,y_r)=E_k(y_r\oplus E_k(y_{r-1}\oplus E_k(y_{r-2}\oplus E_k(\cdots\oplus E_k(y_1\oplus v)\cdots))))$$

其中，$y_i=g_i(x_i)$，$1\leqslant i\leqslant r$。

上述签名方案的合成函数如图 9.1 所示。

2. 环签名的生成 ring-sign$(m,P_1,P_2,\cdots,P_r,s,S_s)$

设签名者为 A_s，要签名的消息为 m，A_s 的私钥为 S_s，全体成员的公钥依次为 P_1，$P_2,\cdots,P_r$，执行以下步骤：

(1) 由消息产生对称密钥 $k=h(m)$ 或 $k=h(m,P_1,P_2,\cdots,P_r)$。

(2) 选取随机的初始值 $v\in\{0,1\}^b$。

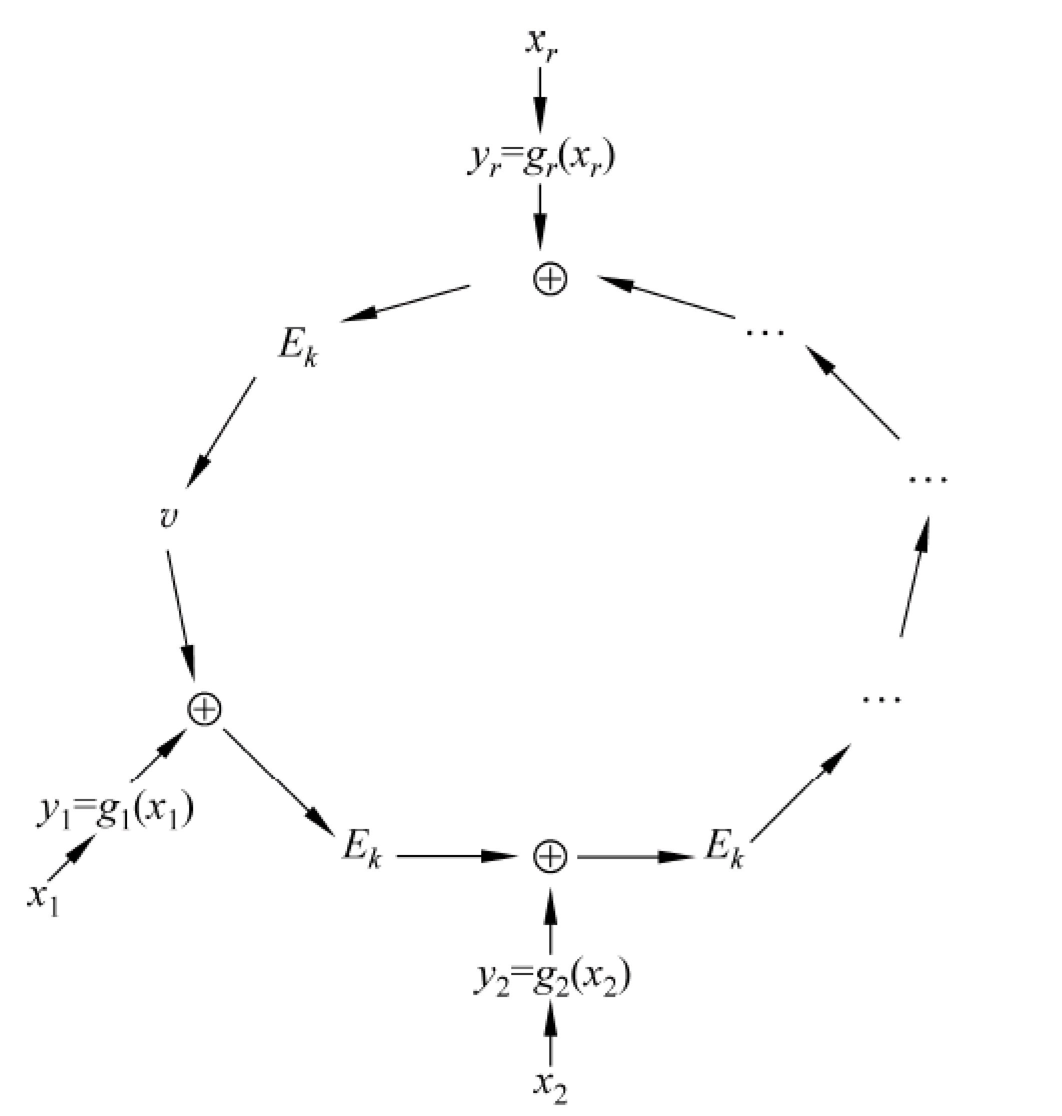


图 9.1 环签名方案合成函数示意图

(3) 对每一个 $i, 1 \leqslant i \leqslant r, i \neq s$,随机选取 $x_i \in \{0,1\}^b$,并计算 $y_i = g_i(x_i)$。

(4) 从方程 $C_{k,v}(y_1, y_2, \cdots, y_r) = v$ 求解出 y_s。

(5) 利用陷门信息(私钥 S_s)求出 $x_s = g_s^{-1}(y_s)$。

(6) 输出环签名 $\sigma = (P_1, P_2, \cdots, P_r, v, x_1, x_2, \cdots, x_r)$。

3. 环签名的验证 ring-verify(m, σ)

对(m,σ),验证者执行以下步骤:

(1) 对每一个 $i, 1 \leqslant i \leqslant r$,利用陷门置换计算 $y_i = g_i(x_i)$。

(2) 计算密钥 $k = h(m)$ 或 $k = h(m, P_1, P_2, \cdots, P_r)$。

(3) 验证方程 $C_{k,v}(y_1, y_2, \cdots, y_r) = v$ 是否成立,若成立,则对消息 m 环签名有效,否则无效。

9.2 指定验证者签名

9.21 指定验证者签名的提出

首先看一个场景:当 Alice 向 Bob 透露自己的个人隐私的时候,或者当 Bob 是招标人而 Alice 是投标人,Alice 向 Bob 递交标书的时候,Alice 希望 Bob 能够验证消息的真实性,不希望 Bob 向第三方证明消息的真实性。换句话说,Alice 希望她发给 Bob 的信息,只有 Bob 能够确信其来自 Alice 并且是完整的,其他任何人都无法判断消息的来源及其完整性,即使 Bob 把自己的私钥泄露给第三方,仍然不能使第三方相信消息是来自 Alice,且未经过篡改。

1996 年的 Eurocrypto 会议上,Jakobsson、Ssko 和 Impagliazzo 提出了指定验证人签

名(designated verifier signature)的概念[①]。指定认证人签名是指签名者把对消息的签名发送给一个指定的验证人，只有这个指定的验证人才能检验签名的有效性。而指定的验证人不能够让其他人相信签名是真实有效的，原因是指定的验证人自己同样可以产生一个有效的签名副本，而且这一副本与原签名人的签名不可区分。这一签名体制在电子商务、电子政务中有很多用途，有效地解决了验证性和隐私性的冲突。

指定验证者签名方案的基本构成如下：

(1) 系统参数和密钥生成。密钥生成中心选取一个安全参数作为输入，生成系统参数并公开。输入安全参数，输入参数方的密钥对$(\mathrm{pk}_i, \mathrm{sk}_i)$$(i=\mathrm{A},\mathrm{B})$，它们分别表示签名者和指定验证者的公钥和私钥。

(2) 指定验证者签名生成。存在一个确定性算法，输入签名者的私钥和验证人的公钥以及待签名消息m，可生成签名$s \leftarrow \mathrm{Sign}(\mathrm{sk}_\mathrm{A}, \mathrm{pk}_\mathrm{B}, m)$。

(3) 指定验证者验证签名。存在一个确定性算法，输入签名者的公钥、验证人的私钥以及消息m，进行验证，若验证通过，返回True，否则为False。即

$$\{\mathrm{True}, \mathrm{False}\} \leftarrow \mathrm{Vrfy}(\mathrm{pk}_\mathrm{A}, \mathrm{sk}_\mathrm{B}, m, s)$$

另外，指定验证者可以独立生成一个签名副本，称为签名模拟。由指定验证人B执行一个算法，当输入签名人A的公钥、B的私钥和一个消息m后，输出B对A的签名的一个模拟，该模拟从概率分布上与A产生的有效签名是不可区分的。

9.2.2 Saeednia-Kremeer-Markowitch 方案

下面通过一个具体方案来举例说明。

S. Saeednia、S. Kremer 和 O. Markowitch 在2003年提出了一个指定验证人签名方案。

(1) 系统参数和密钥生成。p, q为大素数，$q|(p-1)$，g是$\mathbb{Z}_q^*$的一个生成元。$H:\{0,1\}^* \to \mathbb{Z}_q^*$是一个安全散列函数；签名人Alice的私钥$x_\mathrm{A} \in \mathbb{Z}_q^*$，公钥$y_\mathrm{A} = g^{x_\mathrm{A}} \bmod p$。类似地，指定验证人Bob的私钥$x_\mathrm{B} \in \mathbb{Z}_q^*$，公钥$y_\mathrm{B} = g^{x_\mathrm{B}} \bmod p$。

(2) 签名生成。对消息m签名时，Alice随机选取$k, t \in \mathbb{Z}_q^*$，计算

$$c = y_\mathrm{B}^k \bmod p, \quad r = H(m \| c), \quad s = kt^{-1} - rx_\mathrm{A} \bmod q$$

(3) 签名验证。检验Alice对消息m的签名(r, s, t)是否有效，Bob检查下面的等式是否成立：

$$H(m \| (g^s y_\mathrm{A}^r)^{tx_\mathrm{B}} \bmod p) = r$$

(4) 签名模拟。Bob具有签名模拟的能力，如随机选取$s', r' \in \mathbb{Z}_q^*$，计算

$$c = g^{s'} y_\mathrm{A}^{r'} \bmod p$$
$$r = H(m \| c)$$
$$l = r' r^{-1} \bmod q$$
$$s = s' l^{-1} \bmod q$$

① M. Jakobsson, K. Sako, Impagliazzo. Designed Verfier Proofs and Their Applications. Proc. of Eurocrypto96, 1996: 143-154.

$$t = lx_{\mathrm{B}}^{-1} \bmod q$$

把(r,s,t)作为 Alice 对消息 m 的签名的模拟。可见，这里的(r,s,t)与 Alice 发送给 Bob 的对 m 的签名完全不同，但仍然可通过(3)中的签名验证方程。

对本方案有如下解释：

(1) 本方案签名的方式与 Schnorr 签名类似，参数选择也类似。

(2) 签名生成时，用到了 Bob 的公钥。Alice 随机选取两个随机数，为步骤(4)中的签名模拟提供了可能。

(3) 签名验证需要用到 Bob 的私钥，因此其他人无法验证签名。由于 t 是 Alice 随机选择的，且在签名生成方程中使用，所以需要传递给 Bob，用于验证签名。验证函数的正确性是基于

$$(g^s y_{\mathrm{A}}^r)^{tx_{\mathrm{B}}} \bmod p = (g^s g^{rx_{\mathrm{A}}})^{tx_{\mathrm{B}}} \bmod p = g^{(s+rx_{\mathrm{A}})tr_{\mathrm{B}}} \bmod p$$
$$= g^{kx_{\mathrm{B}}} \bmod p = y_{\mathrm{B}}^k \bmod p = c$$

(4) Bob 构造的签名满足签名验证方程。即使 Bob 向第三方透露自己的私钥，也无法证明这个签名是 Alice 签署的，因为 Bob 可根据自己的私钥和 Alice 的公钥独立地生成(r,s,t)。下面验证模拟的签名是正确的：

$$c = g^{s'} y_{\mathrm{A}}^{r'} \bmod p = g^{sl} y_{\mathrm{A}}^{rl} \bmod p = (g^s y_{\mathrm{A}}^r)^{tx_{\mathrm{B}}} \bmod p$$
$$H(m \,\|\, (g^s y_{\mathrm{A}}^r)^{tx_{\mathrm{B}}} \bmod p) = r$$

从而模拟(r,s,t)满足签名验证等式，而且这样的模拟与 Alice 产生的对消息 m 的签名从概率分布上是不可区分的。

思考 9.1 Bob 模拟签名的例子是如何给出的？即如何生成一个有效的(r,s,t)？

实质上是解关于(r,s,t,c)的方程组：

$$r = H(m \,\|\, c)$$
$$(g^s y_{\mathrm{A}}^r)^{tx_{\mathrm{B}}} \bmod p = c$$

先任选一个 $c \in (1, p-1)$，令 $r = H(m \,\|\, c)$，解关于未知数(s,t)的方程$(g^s y_{\mathrm{A}}^r)^{tx_{\mathrm{B}}} \bmod p = c$，这是一个不定方程，由于$(s,t)$均在指数位置，由 c 解出(s,t)有困难。一个巧妙的办法是设法选一个能使得$(g^s y_{\mathrm{A}}^r)^{tx_{\mathrm{B}}} \bmod p = c$ 为恒等式的 c，即避免求解关于(s,t)的方程$(g^s y_{\mathrm{A}}^r)^{tx_{\mathrm{B}}} \bmod p = c$。于是令 $c = g^{s'} y_{\mathrm{A}}^{r'} \bmod p$，求出 $r = H(m \,\|\, c)$，解关于(s,t)的方程$(g^s y_{\mathrm{A}}^r)^{tx_{\mathrm{B}}} = g^{s'} y_{\mathrm{A}}^{r'}$，即有

$$rtx_{\mathrm{B}} = r', \quad stx_{\mathrm{B}} = s'$$

于是先求 $t = r' r^{-1} x_{\mathrm{B}}^{-1}$，再求 $s = s' t^{-1} x_{\mathrm{B}}^{-1}$。

9.3 不可否认签名

9.3.1 不可否认签名的提出

先看一个场景：某公司 A 开发一个软件包，该公司把软件包及其签名卖给用户 B，对于普通的签名，B 可将软件包和签名复制后私自卖给 C。可见普通的签名是不合适的，如果能构造一种签名，在没有 A 的参与及配合下，无法证明签名的真实性和合法性，这样就

防止了B的复制行为，即使B复制了签名，B也无法完成在与C交互的过程中证明签名的真实性和合法性。即没有签名生成方的参与，签名无法证实。

Chaum和van Antwerpen在1989年提出了不可否认签名的概念，使签名者能够限制签名的验证权。该签名可用于特殊的应用场合，如银行用户的保险箱，电子版权保护等。验证签名一般通过"挑战—应答"方式来实现。因此，对于真实的签名，签名人可以通过验证交互过程证明其真实性。另外，签名人对一个真实的签名可能"反悔"，想"否认"这一签名，拒绝参加验证交互过程。为了避免这一情况，签名方案提供了"额外"一个交互过程，用于证明签名是伪造的。如果签名人不参与这个过程，则认为不是伪造的。换句话说，对于伪造的签名，签名人必须通过一个否认交互过程证明其是伪造的。

因此，不可否则签名的本质是：对于真实的签名，必须有签名人参与"验证过程"才能证明其为真，这可防止签名的复制和散布。当然，伪造的签名无法被第三者通过"验证过程"证明为真。另外，为了避免产生的"副作用"，如"后悔的"签名人想通过不参与"验证过程"来否认一个真实的签名是不行的，因为他会被要求参与"否认过程"，通过"否认过程"证明真实签名为假的可能性很小（这便是"不可否认协议"这一名称的由来）。当然，对于伪造的签名，一个签名人可以通过"否认过程"证明其是假的。

于是，不可否认签名由三个部分组成：签名算法，验证协议（即验证过程），否认协议（即否认过程）。

9.3.2 Chaum-van Antwerpen方案

下面介绍经典的Chaum-van Antwerpen不可否认签名方案。

(1) 参数生成。q是一个大素数，$p=2q+1$也是一个大素数，$\mathbb{Z}_p^*$上的离散对数问题是困难的。设$\alpha\in\mathbb{Z}_p^*$是一个阶为q的元素。设$1\leqslant a\leqslant q-1$，令$\beta=\alpha^a \bmod p$。设$G$表示$\mathbb{Z}_p^*$的阶为$q$的乘法子群（$G$其实由模$p$的二次剩余构成）。值$p$、$\alpha$和$\beta$是公钥，$\alpha$是私钥。

(2) 签名生成。对$x\in G$，$y=\mathrm{Sign}(x)=x^a \bmod p$。

(3) 签名验证协议。对于A签署的消息签名对$(x,y)\in G$，执行以下操作：

① B随机选择$e_1,e_2\in\mathbb{Z}_q^*$，计算$c=y^{e_1}\beta^{e_2} \bmod p$。

② A计算$d=c^{a^{-1} \bmod q} \bmod p$发给B。

③ B验证$d\equiv x^{e_1}\alpha^{e_2} \bmod p$，若成立，则签名有效，否则无效。

解释：消息(1)是B生成的"挑战"。消息(2)只有A能正确"应答"，因为"应答"需要使用私钥a，只有A知道。消息(3)验证"应答"的正确性，易知：

$$d\equiv c^{a^{-1}}\equiv(y^{e_1}\beta^{e_2})^{a^{-1}}\equiv x^{e_1}\alpha^{e_2} \bmod p$$

验证协议有一定的错误率。一个伪造的签名（即不是A签署的签名）却被A证明是有效的概率很小，换句话说，在不知道签名私钥的情况下，证明签名有效的概率很小，最多为$1/q$，且该结论不依赖于任何计算假设，即安全性是无条件的。

定理9.1 如果y不是真实的签名，即$y\neq x^a \bmod p$，但B认为y是有效签名，这种情况的概率最多为$1/q$。

证明：A在"应答"时由于不知道私钥a，采取随机猜测一个a^{-1}的办法，易知猜测成

功的概率为 $1/q$。

否认(disavowal)协议用于证明一个伪造的签名是假的,其实由两轮验证协议组成。

(1) B 随机选择 $e_1, e_2 \in \mathbb{Z}_q$,计算 $c = y^{e_1}\beta^{e_2} \bmod p$。

(2) A 计算 $d = c^{a^{-1} \bmod q} \bmod p$ 发给 B。

(3) B 验证 $d \equiv x^{e_1}\alpha^{e_2} \bmod p$,发现不成立。

(4) B 随机选择 $e_1', e_2' \in \mathbb{Z}_q$,计算 $c' = y^{e_1'}\beta^{e_2'} \bmod p$。

(5) A 计算 $d' = c'^{a^{-1} \bmod q} \bmod p$ 发给 B。

(6) B 验证 $d' \equiv x^{e_1'}\alpha^{e_2'} \bmod p$,发现又不成立。

(7) 当且仅当 $(d\alpha^{-e_2})^{e_1'} \equiv (d'\alpha^{-e_2'})^{e_1} \bmod p$ 成立时,B 判断签名 y 是伪造的。

解释:B 在步骤(3)和步骤(6)发现 A 两次"应答"都不对时,主要面临两种可能:一种是签名是伪造的,A 两次都诚实地"应答"(即使用 a^{-1} 进行计算后应答),但都不能验证通过;另一种是签名是真实的,A 两次都故意错误地"应答"(在知道 a 的情况下使用 a'^{-1} $(a' \neq a)$ 来应答),想证明签名是伪造的。这两种可能通过第(7)步来分辨。如果(7)中的等式成立,说明是前一种情况;如果(7)中等式不成立,说明是后一种情况。因此,步骤(7)又叫"一致性检查"。

于是,协议要完成两个目的:

(1) 如果签名是伪造的,则 A 能证明签名是伪造的。

(2) 如果签名不是伪造的(即是 A 签署的),A 想证明签名是伪造的可能性很小(即"不可否认"的可能性很大)。

先讨论第一种情况。由于签名是伪造的,虽然 A 是诚实地"应答",但 A 的两次"应答"都不对,都不能通过验证方程。但是,两次"应答"满足 $(d\alpha^{-e_2})^{e_1'} \equiv (d'\alpha^{-e_2'})^{e_1} \bmod p$,说明 A 知道签名私钥 a,因为 A 成功地约去了 β。具体而言:

$$
\begin{aligned}
(d\alpha^{-e_2})^{e_1'} &\equiv ((y^{e_1}\beta^{e_2})^{a^{-1}}\alpha^{-e_2})^{e_1'} \equiv y^{e_1 a^{-1}}{}^{e_1'}\beta^{e_2 a^{-1} e_1'}\alpha^{-e_2 e_1'} \\
&\equiv y^{e_1 a^{-1} e_1'} \bmod p \\
(d'\alpha^{-e_2'})^{e_1} &\equiv ((y^{e_1'}\beta^{e_2'})^{a^{-1}}\alpha^{-e_2'})^{e_1} \equiv y^{e_1' a^{-1}}{}^{e_1}\beta^{e_2' a^{-1} e_1}\alpha^{-e_2' e_1} \\
&\equiv y^{e_1 a^{-1} e_1'} \bmod p
\end{aligned}
$$

如果 A 不诚实,即 A 想让签名通过验证,必须猜测签署签名的私钥,猜中的可能为 $1/q$,这已经在定理 9.1 中讨论过。

对于第二种情况。签名不是伪造的,但 A 故意错误地回答了两次"应答",想让 B 相信签名是伪造的。即让步骤(7)中的一致性检测通过,等式 $(d\alpha^{-e_2})^{e_1'} \equiv (d'\alpha^{-e_2'})^{e_1} \bmod p$ 满足。这种情况可能性非常小,为 $1/q$。换句话说,不能通一致性检查的概率很高,为 $1-1/q$。

定理 9.2 如果签名真实,且 B 遵守否认协议,A 两次"应答"都不能通过验证方程,即 $d \equiv x^{e_1}\alpha^{e_2} \bmod p$ 不成立,$d' \equiv x^{e_1'}\alpha^{e_2'} \bmod p$ 不成立,但 $(d\alpha^{-e_2})^{e_1'} \equiv (d'\alpha^{-e_2'})^{e_1} \bmod p$ 却是成立的,这种情况的概率很小,为 $1/q$。

证明:显然,A 两次"应答"故意错误地回答了 d 和 d'。即有

$$d \bmod p \neq x^{e_1}\alpha^{e_2} \bmod p$$

$$d' \bmod p \neq x^{e'_1}\alpha^{e'_2} \bmod p$$

但是，又同时有

$$(d\alpha^{-e_2})^{e'_1} \equiv (d'\alpha^{-e'_2})^{e_1} \bmod p$$

下面证明这种情况的概率很小，为 $1/q$。

A 故意用不等于 a^{-1}的两个值来计算应答，粗略地说，两者有 q 种可能，d（以及 d'）的值有 q 种可能，$(d\alpha^{-e_2})^{e'_1} \bmod p$ 和$(d'\alpha^{-e'_2})^{e_1} \bmod p$ 均有 q 种可能，两者相等的概率为 $1/q$。

具体而言，A 可以试图选取不等于 a^{-1}的 r_1, r_2，然后使得两次应答错误，但又要通过一致性检测，于是需要满足如下等式：

$$y^{e_1 r_1 e'_1}\alpha^{e_2 e'_1(ar_1-1)} = y^{e'_1 r_2 e_1}\alpha^{e'_2 e_1(ar_2-1)}$$

令 $y=\alpha^t$，于是有

$$\alpha^{te_1 r_1 e'_1}\alpha^{e_2 e'_1(ar_1-1)} = \alpha^{te'_1 r_2 e_1}\alpha^{e'_2 e_1(ar_2-1)}$$

$$te_1 r_1 e'_1 + e_2 e'_1(ar_1-1) = te'_1 r_2 e_1 + e'_2 e_1(ar_2-1)$$

$$(te_1 e'_1 + e_2 e'_1 a)r_1 - e_2 e'_1 = (te_1 e'_1 + e'_2 e_1 a)r_2 - e'_2 e_1$$

$$r_1 = \frac{(te'_1 e_1 + e'_2 e_1 a)r_2 + e_2 e'_1 - e'_2 e_1}{te_1 e'_1 + e_2 e'_1 a} = \frac{te'_1 e_1 + e'_2 e_1 a}{te_1 e'_1 + e_2 e'_1 a} r_2 - \frac{e'_2 e_1}{te_1 e'_1 + e_2 e'_1 a}$$

A 在选取 r_1, r_2 时，只要不等于 a^{-1}并满足上述关系，即可在两次应答错误的情况下满足一致性条件。显然，B 在不知道 e'_1, e_1, e'_2, e_2, t 的情况下，满足上述等式的概率为 $1/q$。

9.4 失败停止签名

失败-停止(fail-stop)签名的概念是由 B. Pfitzmann 和 M. Waidner 于 1991 年提出的，1992 年 van Heyst 和 Petersen 提出一个具体的方案。它可以防范有强大计算能力的攻击者，一旦该攻击者伪造了一个有效的签名，则签名者马上可以证明攻击者的签名是伪造的，这就是“失败-停止”一词的来历。该签名可用于电子现金系统来防止银行伪造顾客的签名，这对于需要进行大量现金交易的用户而言是十分重要的。

下面介绍 van Heyst 和 Petersen 签名方案。

(1) 参数生成。q 是一个大素数，$p=2q+1$ 也是一个大素数，$\mathbb{Z}_p^*$ 上的离散对数问题是困难的。设 $\alpha \in \mathbb{Z}_p^*$ 是一个阶为 q 的元素。设 $1 \leqslant a_0 \leqslant q-1$，令 $\beta = \alpha^{a_0} \bmod p$。值 p, q, α, β 和 a_0 由可信中心选择，值 p, q, α, β 是公钥，a_0 是对包括签名人在内的人都保密的可信中心的私钥。

签名人的私钥为任选的$(a_1, a_2, b_1, b_2) \in \mathbb{Z}_q$，公钥为$(\gamma_1, \gamma_2)$，其中：

$$\gamma_1 = \alpha^{a_1}\beta^{a_2} \bmod p, \quad \gamma_2 = \alpha^{b_1}\beta^{b_2} \bmod p$$

(2) 签名生成。对于消息 $x \in \mathbb{Z}_q$，定义签名为 $\mathrm{Sign}(x) = (y_1, y_2)$，其中：

$$y_1 = a_1 + xb_1 \bmod q, \quad y_2 = a_2 + xb_2 \bmod q$$

(3) 签名验证。给定消息签名$(x, y_1, y_2) \in \mathbb{Z}_q \times \mathbb{Z}_q \times \mathbb{Z}_q$，

$$\mathrm{Vrfy}(x, y_1, y_2) = \mathrm{True} \Leftrightarrow \gamma_1 \gamma_2^x \equiv \alpha^{y_1}\beta^{y_2} \bmod p$$

为了解释失败-停止的特性，下面依次介绍该签名的特点。

(1) 一个签名公钥对应多个签名私钥。对于公钥(γ_1,γ_2)而言，有 q^2 个私钥(a_1,a_2,b_1,b_2)与之对应。由 $\gamma_1=\alpha^{(a_1+a_0a_2)}$，$\gamma_2=\alpha^{(b_1+a_0b_2)}$，自变量为$(a_1,a_2,b_1,b_2)\in\mathbb{Z}_q$，两个方程为约束条件，于是可变的自变量只有两个。

(2) 给定一个消息签名对$(x,y_1,y_2)\in\mathbb{Z}_q\times\mathbb{Z}_q\times\mathbb{Z}_q$，有 q 个密钥可以满足验证方程。显然，只要满足以下约束条件：

$$
\begin{aligned}
\gamma_1 &= \alpha^{a_1}\beta^{a_2} \bmod p\\
\gamma_2 &= \alpha^{b_1}\beta^{b_2} \bmod p\\
y_1 &= a_1+xb_1 \bmod q\\
y_2 &= a_2+xb_2 \bmod q
\end{aligned}
$$

即可通过验证方程。

简单地说，自变量为$(a_1,a_2,b_1,b_2,a_0)\in\mathbb{Z}_q$，约束条件为 4 个(至少 3 个线性无关，判定留作练习)，故可变的自变量只有一个。

(3) 签名人 Alice 给出一个消息签名对$(x,y_1,y_2)\in\mathbb{Z}_q\times\mathbb{Z}_q\times\mathbb{Z}_q$，敌手猜测出另一个消息 $x'\neq x$ 的签名的概率为 $1/q$。这是(2)的直接推论，因为有 q 个密钥可以满足关于 x 的签名的验证方程，其中只有一个是签名人 Alice 的签名私钥。注意，这一结论是无条件的，不管敌手的计算能力有多大。

(4) 签名人 Alice 发现敌手能给出一个关于消息 x 的不同的签名(x,y_1',y_2')，即不同于 Alice 给出的签名(x,y_1,y_2)，则 Alice 可以立刻证明这个签名是伪造的，于是“停止”。这也称为“伪造证明”(即证明签名是伪造的)算法。

伪造证明的方法是：伪造签名(x,y_1',y_2')通过了验证方程，有 $\gamma_1\gamma_2^x\equiv\alpha^{y_1'}\beta^{y_2'} \bmod p$，$(x,y_1,y_2)$是 Alice 签署的合法签名，有 $\gamma_1\gamma_2^x\equiv\alpha^{y_1}\beta^{y_2} \bmod p$，于是有 $\alpha^{y_1}\beta^{y_2}\equiv\alpha^{y_1'}\beta^{y_2'} \bmod p$，令 $\beta=\alpha^{a_0} \bmod p$，有 $\alpha^{y_1'+a_0y_2'}\equiv\alpha^{y_1+a_0y_2} \bmod p$，即 $y_1'+a_0y_2'\equiv y_1+a_0y_2 \bmod q$，由于$(x,y_1',y_2')$是伪造的，故 $y_2' \bmod q\neq y_2 \bmod q$，因此，$(y_2'-y_2)^{-1} \bmod q$ 存在，于是 Alice 可算出

$$a_0=\log_\alpha\beta=(y_1-y_1')(y_2'-y_2)^{-1} \bmod q$$

Alice 向可信中心表明对 a_0 的拥有，这违反了 Alice 不能计算离散对数问题这一假设，于是证明了消息签名对(x,y_1',y_2')是伪造的。可信中心可以更换 a_0，从而改变 β，签名人的公钥随之改变。

(5) 该签名是一次性签名方案(类似 Lamport 签名方案)，给定一个密钥只能签署一个消息。如果(x,y_1,y_2)，(x',y_1',y_2')使用相同的签名密钥$(a_1,a_2,b_1,b_2)\in\mathbb{Z}_q$(相应公钥为$(\gamma_1,\gamma_2)$)对不同消息 x 和 x' 的签名，则有

$$
\begin{aligned}
\gamma_1 &= \alpha^{a_1}\beta^{a_2} \bmod p & \gamma_2 &= \alpha^{b_1}\beta^{b_2} \bmod p\\
y_1 &= a_1+xb_1 \bmod q & y_2 &= a_2+xb_2 \bmod q\\
y_1' &= a_1+x'b_1 \bmod q & y_2' &= a_2+x'b_2 \bmod q
\end{aligned}
$$

易知，由于 $x\neq x'$，从后面 4 个方程可以解出未知数(a_1,a_2,b_1,b_2)，进而通过前两个方程可以解出 a_0。

除上述外，还有其他高级签名方案，简介如下：

群签名(group signature)：1991 年，Chaum 和 Heyst 首次提出群签名的概念，也称

为组签名。群签名允许组中合法用户以用户组的名义签名,具有签名者匿名的特点。但与环签名的完全匿名性不同,组管理员可以在必要时辨认签名者身份,即具有可追踪性(有限匿名性)。换句话说,由个体代表群体执行签名,验证者从签名不能判定签名者的真实身份,但能通过群管理员查出真实的签名者。

门限签名(threshold signature)。由 Desmedt 和 Frankel 提出,是一种基于"秘密共享"思想的数字签名,它的生成必须由多个成员合作才能完成,但它的验证只需要知道群体的公开密钥即可。Desmedt 等人提出了(t,n)门限签名方案。在该方案中,n 个成员各自拥有整个群体的签名密钥的秘密份额,使得任何多于 t 个成员的子集可以代表群体产生签名,而任何少于 t 个成员的子集则不能产生签名。即为了给消息 m 签名,至少要 t 个组成员合作。当 $t=1$ 时,门限签名就是群签名。当 $t=n$ 时,门限签名就是多重签名。

签名加密(又叫签密,signcryption):一种签名中带有加密的数字签名形式,它的系统和传输开销要小于先签名后加密两者的和。该技术能同时达到签名和加密双重目的。

前向安全(forward security)签名:1997 年的 Eurocrypt 会议上,R. Anderson 提出了前向安全数字签名的概念。主要是考虑了密钥的安全性,签名私钥能按时间段不断更新,而验证公钥却保持不变。攻击者不能根据当前时间段的私钥推算出先前任何一个时间段的私钥,从而不能伪造过去时间段的签名,对先前的签名进行了保护。这种思想能应用到各种类型的签名中,提高系统的安全性。

变色龙签名(chameleon signature)是 H. Krawczyk 和 T. Rabin 于 2000 年引入的一种特殊的数字签名。先对消息散列,后进行签名。其名字来源就是使用了名为 Chameleon 的散列函数。Chameleon 散列函数具有一个公钥和对应的私钥,通常称该私钥为陷门信息。散列函数具有如下性质:任何掌握其公钥的人都可以使用该散列函数来计算散列值,在不掌握陷门信息的情况下,该散列函数是抗碰撞的;但掌握陷门信息后,给定一个散列值输出,容易找到另外一个输入,使其散列值相等。因此,签名者使用接收方的 Chameleon 散列函数 H,即 H 的陷门信息只有接收方知道。那么,签名具有如下性质:签名者不可抵赖;接收方不可能把签名出示给第三方而使第三方相信。因为接收方掌握陷门信息,可以轻易地产生冲突信息,所以其他人都不能确定该签名是否由签名者所签。由于散列函数 H 是接收方的,故同一消息对不同的接收方要使用不同的散列函数,所以对不同的接收方都要进行签名,而不是只签名一次。

传递签名(transitive signature):由 Micali 和 Rivest 在 2002 年首次提出,实际上是一种特殊的具有同态特征的签名,主要用于对具有传递性的二元关系进行高效签名。考虑一个表示可传递二元关系的图 $G(V,E)$,传递签名是这样的一个签名方案,签名者 Alice 对图 G 的边进行签名,使得任何人只要得到 Alice 对边(u,v)和(v,w)的签名,无须知道签名私钥就可简单地计算出关于边(u,w)的签名,而且这样产生的签名与由 Alice 产生的签名不可区分。

第 2 部分

小　结

本部分介绍了数字签名知识的扩展。主要是具有附加功能的签名方案。本部分涉及的知识点如下所示：

第 2 部分的主要知识点
- 盲签名
  - Chaum 盲签名 → Schnorr 盲签名 → Neberg-Rueppel 盲签名
  - ElGamal 盲签名　ElGamal 型盲签名的一般构造方法　盲签名的应用
- 代理签名
  - 不保护代理的代理签名
  - 保护代理的代理签名
- 多重数字签名
  - 广播多重数字签名
  - 顺序多重数字签名
- 环签名
- 制定验证者签名——Saeednia-Kremeer-Markowitch 签名方案
- 不可否认签名——Chaum-van Antwerpen 签名方案
- 失败停止签名
- 其他数字签名
  - 群签名
  - 门限签名
  - 签密
  - 前向安全签名
  - 变色龙签名
  - 传递签名

本部分的重点是盲签名、代理签名和多重签名，难点是不可否认签名和失败停止签名。

扩展阅读建议

更多关于签名的集中论述可以参见如下参考文献。

赵泽茂.数字签名理论.北京：科学出版社，2007.（各种具有附加属性的签名的集中介绍）

曹正军，刘木兰.唯签名的数字签名模型的分类.中国科学 E，2008，38(2)：223-235.（讨论签名模型的分类）

曹珍富.Classification of signature-only signature models. http://eprint.iacr.org/2006/164，2006-05.

Guilin Wang. Bibliography on digital signatures. http://www.uow.edu.au/~guilin/bible.htm.

H. Lipmaa. Cryptographic signature schemes. http://www.adastral.ucl.ac.uk/helger/crypto/link/signature.

第 3 部分

安 全 协 议

1. 密码协议的概念

密码协议(cryptographic protocol)是以密码算法为基础的协议,也称为安全协议。协议是指双方或者多方为完成一项任务所进行的一系列有序步骤。因此,一个人执行一系列步骤不构成协议;必须完成一项任务,否则不是协议。步骤是有序的,每一步必须依次执行,前一步完成之前,后一步不能执行。协议一般具有以下特点:

(1) 协议中的每一方都必须了解协议,并且预先知道所要完成的所有步骤。

(2) 协议中的每一方都必须同意并遵循它。

(3) 协议必须是清楚的,每一步必须明确定义,并且不会引起歧义。

(4) 协议必须是完整的,对每种可能的情况必须规定具体的动作。

下面是一个简单的分苹果协议。有两方参与者 A 和 B,完成公平分苹果的任务,执行如下协议: ①A 将苹果切成两份; ②B 选择其中一份作为自己应得; ③A 取剩下的一份作为自己应得。该协议也称为“分割-选择”协议。

2. 构成单元

密码协议的基本构成单元是密码算法,主要包括 4 个方面:

(1) 公钥密码算法,在分布式环境中实现高效的密钥分发、密钥协商和实体认证。

(2) 对称密钥算法,高效率的实现信息的保密性。

(3) 散列函数,实现协议中的消息完整性。

(4) 随机数发生器,为每个参与者提供随机数。

3. 分类

协议可从以下几个方面进行分类:

(1) 按协议的轮数,分为 2 轮协议、3 轮协议……n 轮协议。

(2) 按协议的功能,分为身份识别协议、实体认证协议、密钥分配协议、密钥协商协议、身份识别与密钥建立协议、秘密共享协议、不经意传输协议和电子商务协议等。

(3) 按协议应用的目标,分为选举协议、拍卖协议和支付协议等。

(4) 按协议的交互性,分为交互协议和非交互协议。

(5) 按协议的第三方性质,分为仲裁协议、裁决协议和自动执行协议。

(6) 按协议的复杂性,分为基础协议(如比特承诺协议)、中级协议(如零知识协议)、高级协议(如安全多方计算协议)。

4. 密码协议的设计原则

如果能在密码协议的设计阶段就充分考虑到一些不当的协议结构可能威胁协议的安全性，并加以避免，将事半功倍。Martin Abadi 和 Roger Needham 提出了设计协议应当遵守的一些原则。

(1) 消息独立完整性原则。协议中的每条消息都应能够准确地表达出它所想要表达的含义。一条消息的解释应完全由其内容来决定，而不用借助于上下文来推断。假设协议的一个步骤是 A→B：M，想表达 A 把消息 M 发给 B。但因为消息 M 没有与 A 和 B 绑定，光从 M 看不出是由 A 发给 B 的，要借助上下文来分析。这样攻击者可能利用这一点发起攻击。

(2) 消息前提准确原则。与消息相关的先决条件应当明确给出，并且其正确性与合理性应能得到验证。这一原则是在上一原则基础上的进一步要求，不仅要考虑消息本身，还要考虑与每条消息相关的条件是否合理，每条消息所基于的假设是否能够成立。

(3) 主体身份标识原则。如果一个主体的标识对于某个消息的含义是重要的，就应当在消息中明确地附加上主体的名称。主体的名称可以是显式的，即以明文形式出现，也可以是隐式的，即采用加密或签名技术对主体名称进行保护。

(4) 加密目的原则。明确加密的目的，否则将造成冗余。密码算法的不正确应用可能导致协议出现错误。因此，应用密码算法时，必须知道使用的理由和使用的正确方法。

(5) 签名原则。签名可以确保数据的真实性和抗抵赖性。如果需要同时采用签名和加密，应当采用先签名后加密的方式。应当对数据的散列值进行签名，不直接对数据签名。

(6) 随机数的使用原则。在协议中使用随机数可以提供消息的新鲜性。但是在使用随机数时，应当明确其所起的作用和属性。随机数应当保证良好的随机性。

(7) 时间戳的使用原则。当使用时间戳时，必须考虑各个计算机的时钟与标准时钟的误差，这种误差不应当影响协议执行的有效性。时间戳的应用十分依赖系统中时钟的同步，但是这在实际中是不容易做到的。

(8) 编码原则。协议中消息的编码格式与协议安全密切相关，应当明确协议中消息的具体数据格式，而且还要验证这种格式对安全的贡献。

(9) 最少安全假设原则。在进行协议设计时，常常要对系统环境进行风险分析，做出适当的初始安全假设，如认为所采用的密码算法是安全的，认证服务器是可信的等。但是，初始的安全假设越多，协议的安全性就越差。这是因为，一旦初始的安全假设的安全性受到威胁，将直接影响协议的安全性。因此，协议设计时应当采用最少安全假设原则。

5. 密码协议的安全分析

协议安全分析的目的就是揭示协议是否存在安全漏洞和缺陷，主要采用形式化的分析方法，即将协议的描述形式化，借助于人工和计算机分析推理来判断协议是否安全。它不仅能够发现已知的安全漏洞和缺陷，还能发现未知的安全漏洞和缺陷。协议的形式化分析方法主要有 3 类：

(1) 形式逻辑方法。它是一种基于知识的推理分析方法。以 BAN 逻辑为代表，运用推理规则进行分析，如最终的知识语句集合中不包含所要得到的目标知识语句时，就说明

协议存在安全缺陷。

（2）模型检测方法。基本思想是把协议看成一个分布式系统，每个主体执行协议的过程构成局部状态，所有局部状态构成系统的全局状态。每个主体的收发动作都会引起局部状态的改变，从而也就引起全局状态的改变。在系统可达的每一个全局状态检查协议的安全属性是否得到满足，如果不满足，则检测到协议的安全缺陷。模型检测方法已被证明是一种非常有效的方法，具有自动化程度高，检测过程不需要人工参与，协议存在安全缺陷时能自动产生反例等特点。但由于该方法采用穷举搜索存在攻击的所有可能的执行路径，容易产生状态空间爆炸问题。

（3）定理证明方法。该方法试图证明协议满足安全属性，而不是寻找对协议的攻击。代表性的方法有SPI演算方法、归纳方法、串空间方法等。

第 10 章 实体认证协议

本章介绍用于允许一方(验证者)得到另一方(声称者)所声称的“实体”的保证技术,由此可防止身份假冒。例如,当用户 A 登录到计算机(或者自动取款机、电话银行等)时,计算机怎么知道他不是由其他人假冒的呢?特别是在开放的网络环境下,该问题尤为突出。验证者最常用的技术是检测用来证明声称者拥有和真实方相关联的秘密的正确性。这种计算包括身份识别(identification),实体认证(entity authentication)和身份验证(identity verification)。与身份识别相关联的概念包括消息源鉴别(data-origin authentication,如基于对称密码的消息鉴别码和基于公钥的数字签名)和认证的密钥建立(authenticated key agreement,即密钥建立之前先验证密钥建立双方实体的真实性)。

值得思考的一个问题是:实体认证与消息鉴别有何异同。

实体认证和消息鉴别的主要不同在于:消息鉴别在产生消息时,本身不提供时效性保证,而实体认证一般是实时的,能保证协议执行时确认声称者实体。实体认证通常证实实体本身,而消息鉴别除了鉴别消息源和验证完整性外,通常关心消息的具体内容。换言之,实体认证所涉及的通常是没有意义的消息,消息鉴别涉及的是特定实体所声称的有意义的消息。认证的密钥建立协议包括实体认证协议和密钥建立协议两个部分,密钥建立本质上是消息认证,其中消息就是密钥。实体认证和消息鉴别的联系在于,在实体认证协议中将实体的身份当作一条消息来处理,这样就可以使用消息鉴别机制来实现实体认证,这也是构造实体认证协议的一种合适的方法。

本章首先介绍实体认证与身份识别的概念;然后介绍基于口令的实体认证,基于“挑战应答”协议的实体认证,主要利用对称密码加密、公钥密码加密、散列函数等基本模块来构造协议;最后介绍身份识别协议,主要基于零知识证明的方法来构造。

10.1 实体认证与身份识别概述

10.1.1 实体认证的基本概念

身份识别协议的一般设置涉及声称者(或证明者,claimant)A 和验证者(verifier)B。验证者表现或预先假定为声称者所声称的身份。目的是确认声称者的身份的确是 A,即提供实体认证。

定义 10.1 实体认证是一个过程,即其中一方(通过获得证实的证据)确信参与协议的另一方的身份,并确信另一方真正参与了该过程(即在证据获得之时或之前是活动的)。

实体认证的基础。实体认证根据认证的依据可分为三大类：

（1）已知的事物（knowledge）。例如口令、个人识别码 PIN 以及在挑战应答协议中已被证实的秘密或私钥。

（2）已拥有的事物（possession）。通常是物理器具，如磁卡、芯片卡、智能卡以及提供按时间变化的口令的手持式定制计算机器（口令生成器）。

（3）固有事物（characteristics）。包括利用人类生理特征的方法，如手写签名、指纹、声音、虹膜、手掌纹等特征。这些技术通常是非密码学的，这里不作进一步讨论。

10.1.2 身份识别的基本概念

1. 身份识别协议的目的

从验证者的角度，身份识别协议的结果或者接受声称者的身份是可信的（完全接受），或者是终止接受（拒绝）。具体而言，身份识别协议的目的有以下 3 个：

（1）就诚实方 A 和 B 而言，A 可成功地向 B 认证他自己，即 B 完成协议接受 A 的身份。

（2）不可传递性（non-transferability）。B 不能重新使用和 A 交换的身份识别协议来成功地向第三方 C 假冒 A。

（3）假冒（impersonation）。任何不同于 A 的实体 C 执行协议并担当 A 的角色，使得 B 完成协议接受 A 的身份的概率可忽略。

上述几点永真，即使 A 和 B 之间大量（多项式数量）的认证被观察到；敌手 C 参与了与 A 和 B 一方或双方的上述协议的执行；由 C 发起的协议的多个实例可能同时运行。

2. 身份识别协议的性质

身份识别协议有许多特性，通常包括以下几个：

（1）身份识别的交互性。一方或双方可以确认其他方的身份，分为单向认证或者双向认证。

（2）计算效率。执行协议所需要的计算量。

（3）通信效率。包括传输（消息交换）的步数和需要的带宽（总传输的比特数）。

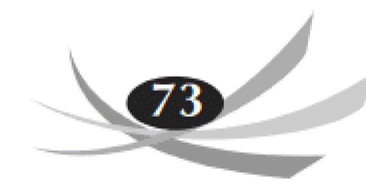
3. 身份识别与实体认证的区别

身份识别和实体认证的含义基本相同，很多地方没有严格区分两者，其含义略有区别：身份识别只是声称身份，而实体认证则是确认身份。两者之间有着微妙的关系：

（1）从概念上看，身份识别是一种声称自己身份的行为，而实体认证是验证所声称身份真实的过程。前者强调声称，后者强调验证。身份识别中，所声称的身份信息是公开的；而在实体认证中，交互双方可能会用到只有他们才知道的共享秘密信息（如口令等）。

（2）从能抵抗的威胁来看，对于实体认证，当声称者和验证者共享秘密需要合作的时候，认证协议只能抵抗来自外部的威胁，一般不考虑内部攻击，即验证者不诚实的情况；身份识别协议可以考虑来自内部的攻击，即验证者可以是不诚实的。

（3）假设两类协议都是安全的，则从协议执行结束时的效果来看，对于实体认证，声称者 A 能够向验证者 B 证明自己确实是 A，设计目标是：其他人则无法冒充 A 使 B 相信其正在和 A 会话（但 B 可以假冒 A）；对于身份识别，声明自己身份的 A 可以使验证者 B



相信他确实是 A，但 B 事后无法使其他人相信自己是 A，即无法成功假冒 A。即设计目标是：身份可识别，且不透露任何可被将来用于身份假冒的信息。

(4) 从使用场合来看，实体认证有时会结合密钥交换一起使用以产生一个经过认证的会话密钥，而身份识别则一般不考虑具体目的。

10.1.3　对身份识别协议的攻击

对身份识别协议的攻击包括如下方面：

(1) 假冒。一个实体声称是另一个实体。

(2) 重放攻击。针对同一个或不同的验证者，使用从以前执行的单个协议得来的信息进行假冒或其他欺骗。

(3) 交织攻击。对从一个或多个以前的或同时正在执行的协议（并行会话）得来的消息进行有选择的组合，从而假冒或进行其他欺骗，其中的协议包括可能由敌手自己发起的一个或多个协议。

(4) 反射攻击。从正在执行的协议将消息发送回该协议的发起者的交织攻击。

(5) 强迫攻击。敌手截获一个消息（一般包括一个序列号），并在延迟一段时间后重新将该消息放入协议，是协议继续，此时强迫延时发生。

(6) 选择挑战攻击。是对挑战-应答协议的攻击，其中敌手有策略地选择挑战以尝试提取声称者的长期密钥的信息。选择挑战攻击是指将声称者作为一个预言机（oracle，也有文献称之为谕示机），即获得的信息不能单独从声称者的公钥计算出来。假如协议要求声称者对挑战进行加密或者计算消息鉴别码（MAC），则攻击可能包含选择明文攻击；假如协议要求声称者对挑战进行签名，则攻击可能包含选择消息攻击；假如协议要求声称者对挑战进行解密，则攻击包含选择密文攻击。表 10.1 给出了对身份识别协议的攻击及其应对措施。

表 10.1　对身份识别协议的攻击及其应对措施

攻击类型	避免攻击的原理
重放	用挑战-应答技术；使用临时值；在应答中加入目标身份
交织	协议运行过程中链接所有信息（使用链接的临时值，例如临时值 r 递增）
反射	在挑战-应答协议中嵌入目标实体的标识符；用每个不同形式的消息构造协议（避免消息的对称性）；单向密钥的使用
选择挑战	用零知识技术，在每个挑战-应答中嵌入自选择的随机数（混淆者）
强迫延时	随机数与短应答超时结合使用；时间戳加上适当的附加技术

10.2　基于口令的实体认证协议

传统的基于口令的认证协议（Password Authentication Protocol，PAP）也称为弱认证，口令简单易记，因而是一种使用广泛的认证技术，特别适用于用户远程访问计算机系

统的模式。通常在通信连接建立的阶段进行 PAP 认证。由于其使用方便,费用低廉,因此操作系统(如 UNIX、Windows NT、NetWare 等)都提供了对基于口令的认证的支持。在这种类型的认证中,用户和计算机共享某个口令,这个口令相当于一个长期使用但又相对较短的对称密钥。如果用户 U 希望使用主机 H 的服务,H 必须事先对 U 进行初始化,发给 U 一个口令 PWD_U,或者 U 自己选择一个口令。本章的编排采用层层递进的方式,以便于读者理解其中的协议设计的演变过程和动机。

10.2.1 基于口令的认证协议

一个最简单的形式就是主机 H 在初始化用户 U 之后,建立一个保存所有用户口令的文档(称为口令表),其中每一条记录形如(ID_U,PWD_U),分别对应用户的身份和该用户的口令。用户每次登录 H 时,H 都要求其输入口令,然后从自己存储的用户口令表中查找以判定 U 的输入是否正确。过程如下:

1. U→H:ID_U
2. H→U:"Input Your Password"
3. U→H:PWD_U
4. Check PWD_U in Table

该协议非常简单。在 20 世纪 70 年代,由于终端和主机之间的通信链路是不可攻击的专线,所以该协议在当时的环境下确实能够提供从用户到主机的认证。但是该协议不适用于现在的网络通信环境:由于身份和口令均为明文传输,故在开放信道上容易被窃听者窃听,窃听者可用该身份发起认证协议,并利用该口令完成认证过程,从而得到 H 的服务。另外,主机 H 中的口令明文存储也增加了口令泄露的可能,如果系统攻击者窥探了口令表(如通过利用软件漏洞渗透到主机系统,得到管理员权限,留下后门程序,发送口令表给攻击者),则所有用户的口令均泄露。因此,这里主要面临两种敌手模型,一个是窃听者,另一个是系统攻击者(还可进一步划分为终端系统攻击者和主机系统攻击者)。

思考 10.1 如何解决口令表泄露问题?

引入单向函数的口令认证协议如图 10.1 所示。

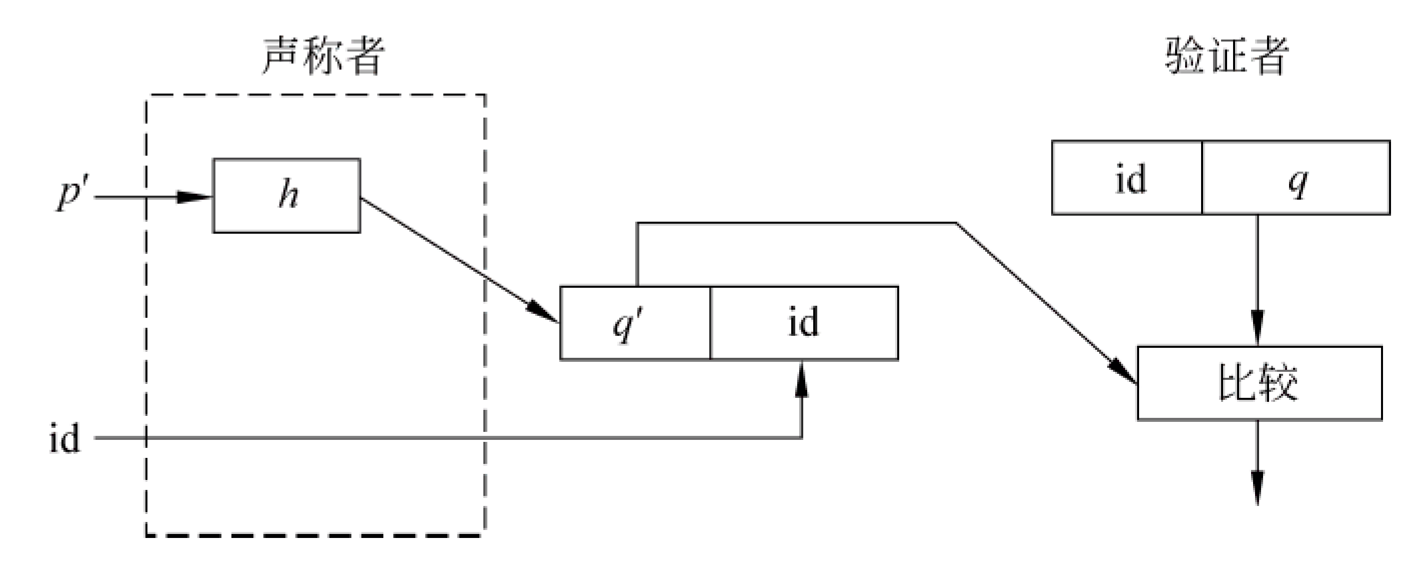
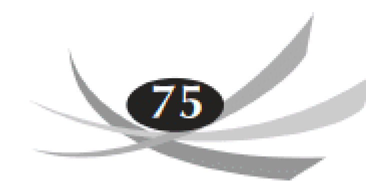


图 10.1 引入单向函数的口令认证协议

为了解决上一个协议中的主机口令表中的口令泄漏问题,防御系统攻击者,可以采取的办法是将口令进行单向函数(例如散列函数)计算,然后保存到口令表中。即口令表中保存的是(ID_U,OWF(PWD_U)),这里 OWF()表示单向函数。这是可行的,因为主机只需



要区分有效口令和无效口令，无须知道口令本身。即使口令表泄露，由于单向函数的性质，从 OWF(PWD_U)得不到 PWD_U。该改进措施也可以抵御在线窃听者，因为在线窃听者只能窃听到散列值。但是由于口令的长度限制为便于记忆的长度范围，例如6位，则攻击者可能在得到口令表后，离线尝试所有可能的密钥，试图找到一个满足 OWF(PWD_U)的 PWD_U，该攻击方式称为字典攻击(dictionary attack)。或者进行在线字典攻击，即在线尝试对口令的猜测集合(称为字典)。

思考 10.2 为了消除字典攻击，可采取什么方法?

可以采用图10.2所示的加 Salt 机制的认证，其目标是增加口令熵，即使用不易猜测的口令。另一个增加口令熵的方法是将口令表设计成(ID_U, salt, OWF(PWD_U, salt))，这样，如果 salt 足够大，在进行在线字典攻击时不容易找到 PWD_U，因为要尝试所有可能的 salt。同时，虽然 salt 是明文存储，没有增加穷举的计算量，但因为 PWD_U 和 salt 前后组合的方式是未知的，穷举 PWD_U 找到满足 OWF(PWD_U, salt)的难度会增大，使得离线字典攻击的难度加大。另外，也使得攻击者无法进行成批量的口令猜测。

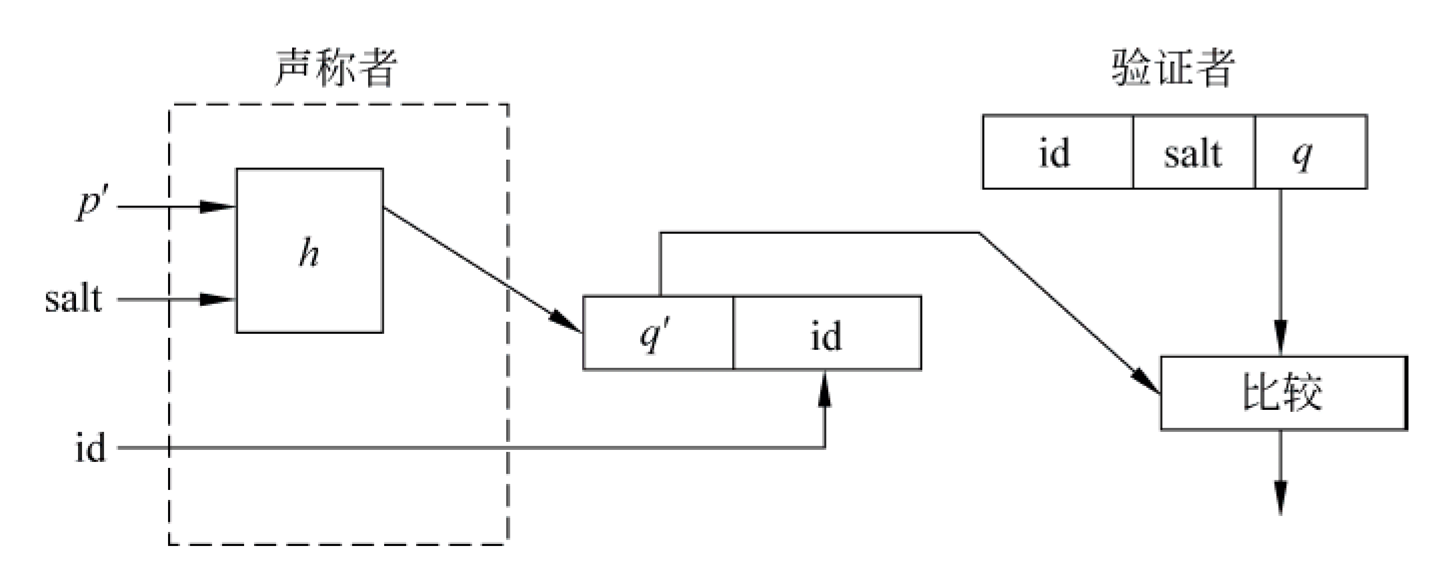


图 10.2 加 Salt 机制的认证示意图

对于抵抗在线字典攻击，通常的办法是限制尝试口令次数、延迟响应或者 CAPTCHA(Completely Automated Public Turing Test to Tell Computers and Humans Apart，全自动区分计算机和人类的图灵测试)方法。CAPTCHA 系统使用人工智能难题的图形码技术，服务器端生成随机字符串 R，经变换成 $F(R)$后发送给用户，由于机器无法识别变形图片中的字符串 R，只有人才能识别，保证了客户端参与者必须是人，因此避免了攻击者利用机器进行自动的在线字典攻击。

10.2.2 基于散列链的认证协议

为防御在线口令窃听者和重放攻击，L. Lamport 在20世纪80年代首次提出一个简单的动态口令方法，也称为强认证和一次口令(one-time password)方案。该方法描述如下：

主机保持用户的初始口令为(ID_U, n, $Hash^n(PWD_U)$)，其中 n 是一个较大的数，如1000，Hash()为一个散列函数，$Hash^n(PWD_U)=Hash(\cdots Hash(PWD_U))\cdots)$。用户 U 只需记住口令 PWD_U，每次用户 U 登录时，H 都会更新自己保存的用户 U 的口令记录。H 和 U 首次运行口令认证协议时，用户端对 PWD_U 重复计算散列函数 $n-1$ 次，得到 $Hash^{n-1}(PWD_U)$，计算结果发送给 H，H 对该结果执行一次散列运算后，与保存的 $Hash^n$

(PWD_U)进行比较，如果相同则认证通过。然后将$(ID_U, n, Hash^n(PWD_U))$更新为$(ID_U, n-1, Hash^{n-1}(PWD_U))$。如此反复，在第 c 次认证过程中，发送 $Hash^{n-c}(PWD_U)$，接收者接收后，进行一次散列运算，与保存的 $Hash^{n-c+1}(PWD_U)$进行比较，如果相同则认证通过，然后将保存的 $Hash^{n-c+1}(PWD_U)$更新为 $Hash^{n-c}(PWD_U)$。

例 10.1 这一设计思想可应用到 S/KEY 认证协议中。

图 10.3 给出了 S/KEY 一次性口令认证的示意图。

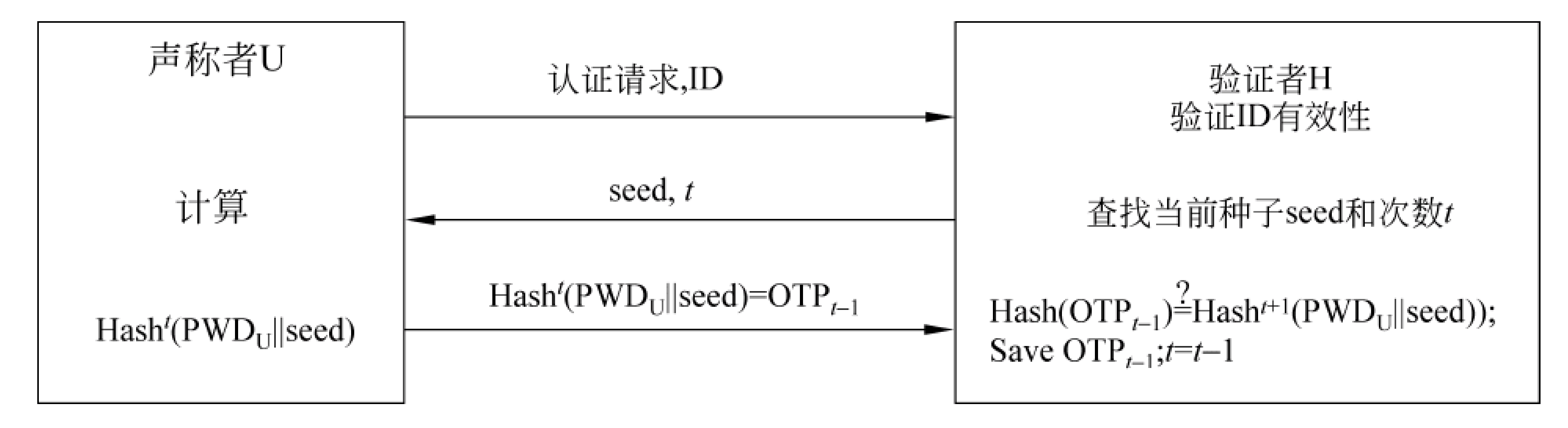


图 10.3 S/KEY 一次性口令认证示意图

S/KEY 认证过程描述如下：

1. $U \to H: ID_U$
2. $H \to U: seed, t$
3. $U \to H: Hash^t(PWD_U) \| seed) = OTP_{t-1}$
4. $Hash(OTP_{t-1}) \stackrel{?}{=} Hash^{t+1}(PWD_U \| seed))$;

Save $Hash^t(PWD_U \| seed)$; $t = t-1$

当计数器 c 的值从 n 递减到 1 时，用户 U 和主机 H 需要重新初始化以设置口令。这种一次口令系统也称为 S/KEY，已经成为标准的协议 RFC1760。散列函数可以选择 MD5 算法或者 SHA1 算法。由于计数器的存在，有利于保持用户主机间的同步，但是攻击者可能利用传输中的计数器发起中间人攻击：攻击者将计数器改小发送给用户，得到用户的应答后，可用于计数器较大的应答，从而得到该用户一系列的有效口令，进而在一段时间内假冒合法用户而不被觉察（这就是所谓“小数”攻击），这一缺陷的来源是 S/KEY 认证协议中缺乏完整性保护。

例 10.2 这一设计思想可应用到基于口令的智能卡认证协议。

首先是注册协议，如图 10.4 所示。

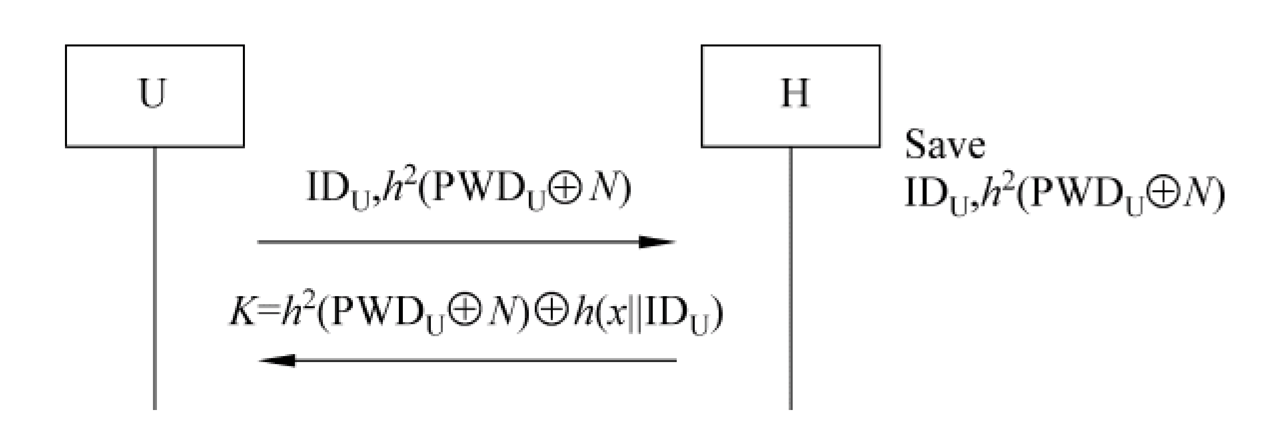
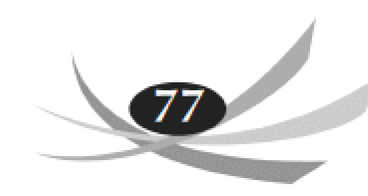


图 10.4 智能卡注册协议

设计中，U 使用随机选取的 N 增加了 PWD_U 的口令熵，共享密钥为 H 随机选取的 x



连接 ID_U，然后运用散列函数，与保存的口令散列值异或。

基于口令的智能卡认证协议如图 10.5 所示。

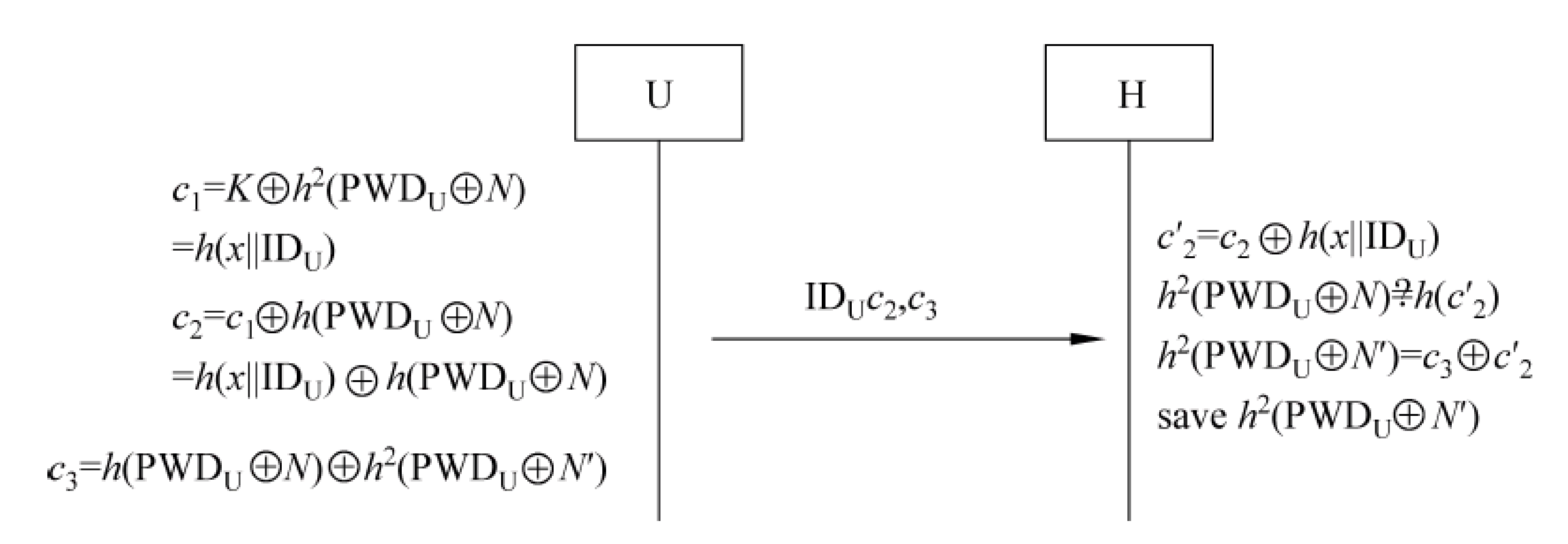


图 10.5　基于口令的智能卡认证协议

容易观察认证等式的正确性。认证完成后，对保存的口令散列值进行了更新。

10.2.3　基于口令的实体认证连同加密的密钥交换协议

基于口令的认证协议可用于进行加密的密钥交换(Encrypted Key Exchange，EKE)，Bellovin 和 Merritt 在 1992 年设计了一个基于口令认证的 EKE(见图 10.6)。在该协议中，用户 U 和计算机 H 共享口令 PWD_U，这个口令短小、易于记忆。另外，用户和计算机事先约定一种对称密钥加密体制 SE 和一种公钥加密体制 AE，$SK_K(\cdot)$表示用密钥 K 进行对称加密，$AE_{PK}(\cdot)$表示用公钥 PK 进行公钥加密。协议如下：

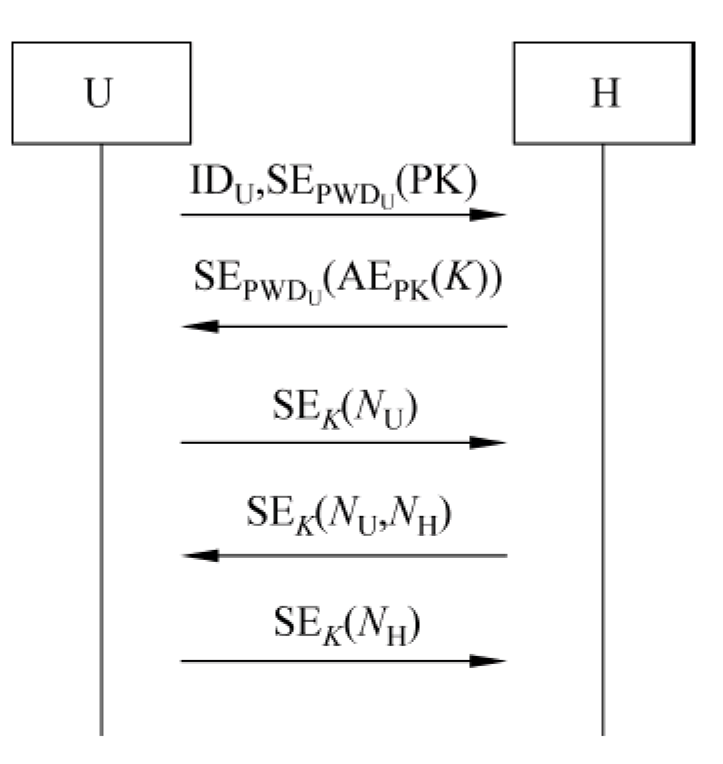
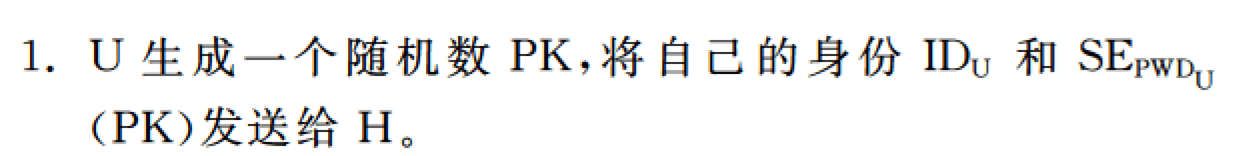
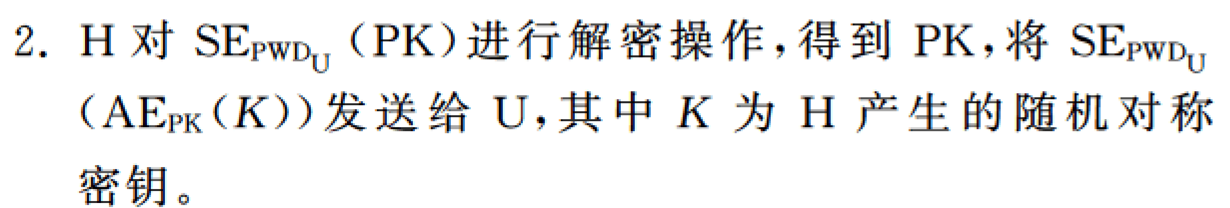


图 10.6　加密的密钥交换协议



1. U 生成一个随机数 PK，将自己的身份 ID_U 和 $SE_{PWD_U}(PK)$发送给 H。



2. H 对 $SE_{PWD_U}(PK)$进行解密操作，得到 PK，将 $SE_{PWD_U}(AE_{PK}(K))$发送给 U，其中 K 为 H 产生的随机对称密钥。
3. U 利用 PK_U 和 PK 从 $SE_{PWD_U}(AE_{PK}(K))$中恢复出 K，将 $SE_K(N_U)$发送给 H，其中 N_U 为用户 U 选取的随机数。
4. H 从 $SE_K(N_U)$中恢复出 N_U，将 $SE_K(N_U,N_H)$发送给 U，其中 N_H 为 H 选取的随机数。
5. U 从 $SE_K(N_U,N_H)$中恢复出 N_H，将 $SE_K(N_H)$发送给 H。
6. 如果 H 能够从 $SE_K(N_H)$中恢复出 N_H，则认证通过，并使用 K 作为共享密钥进行安全通信。

容易看出，步骤 1 是通过共享的口令传送一个随机数 PK，使得密钥信息熵扩大。步骤 2 类似步骤 1，传递一个随机数 K 作为后续步骤的共享密钥，进一步扩大了共享密钥的信息熵并且保证了密钥的“新鲜性”。因此，步骤 1 和步骤 2 可视为两次对 PWD_U 的加盐(salt)操作，使窃听者看到的数据与 PWD_U 保持统计独立。另外，步骤 3～5 是基于对称密钥的双向认证协议，可以用任意基于对称密钥的双向认证协议替代。

图 10.7 给出以上构造的具体实例——基于 Diffie-Hellman 的 EKE 协议。P 是一个大素数，至少 1024 位。

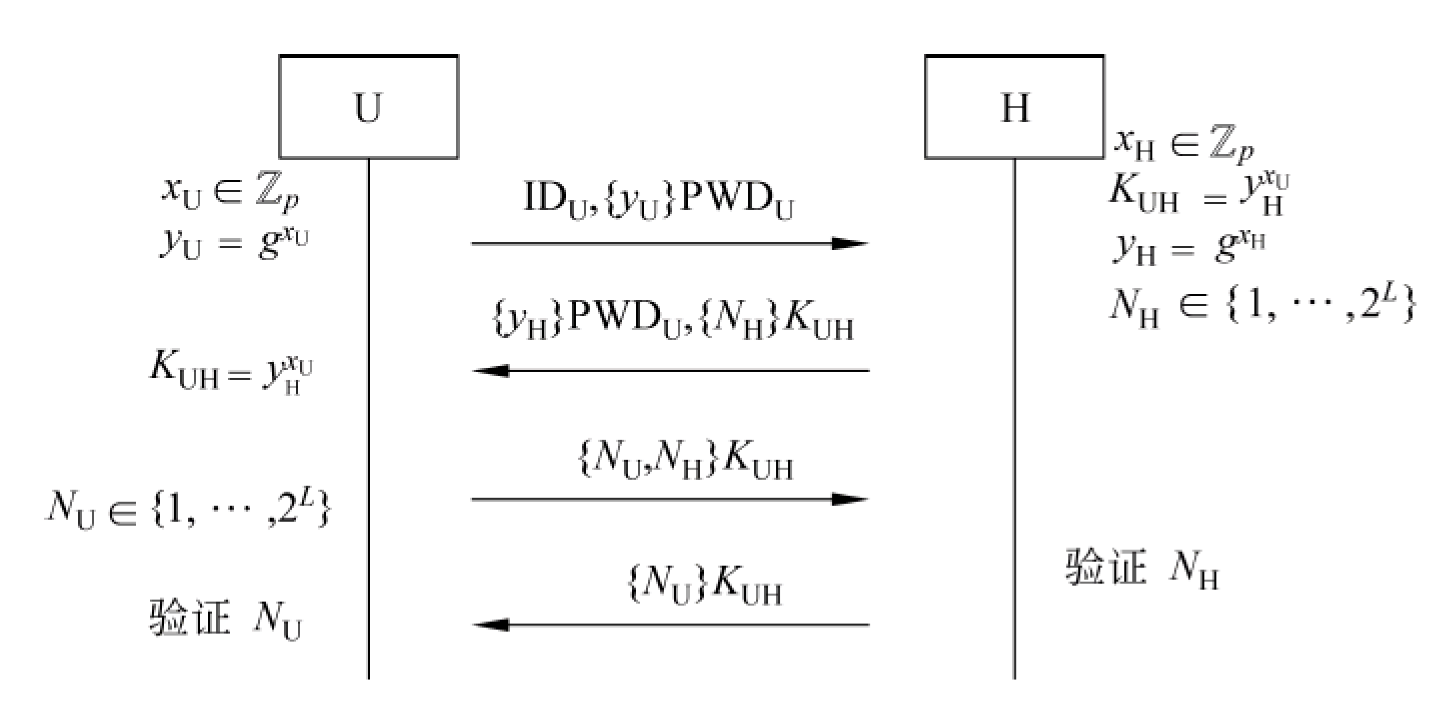


图 10.7　基于 DH 的 EKE 协议

10.3　基于“挑战-应答”协议的实体认证

10.3.1　基于对称密码的实体认证

基于“挑战-应答”协议的认证也称为强认证。国际标准 ISO/IEC 9798—2 详述了 6 个使用对称密码算法的协议，其中有 4 个协议仅用来提供实体认证，另外两个提供实体认证和密钥建立。4 个认证协议中，有两个是单向认证协议，两个是双向认证协议。约定通信双方为 A 和 B，$\{M\}_K$ 表示密钥 K 加密消息 M，并提供完整性保护。

协议 1：一路(one pass)单向(one-way)认证协议。

一路表示一条消息，发起者 A 发送一条消息，验证者 B 验证 A 的身份，为单向认证。B 由时间戳 T_A 判断 A 是否有效，包含身份标识 B 能保证 A 知道 B 是期望的验证实体。

$$A \rightarrow B: \{T_A, B\}_{K_{AB}}$$

协议 2：两路单向认证协议。

协议 2 与协议 1 类似，只是用随机数替代时间戳。

$$B \rightarrow A: N_B$$
$$A \rightarrow B: \{N_B, B\}_{K_{AB}}$$

协议 3：两路双向认证协议。

思考 10.3　能否利用协议 1 设计该协议？

协议 3 由协议 1 的两个实例组成，提供双向认证。

$$A \rightarrow B: \{T_A, B\}_{K_{AB}}$$
$$B \rightarrow A: \{T_B, A\}_{K_{AB}}$$

协议 4：三路双向认证协议。

协议 4 与协议 3 类似，提供双向认证，只是用随机数代替了时间戳。

$$B \rightarrow A: N_B$$
$$A \rightarrow B: \{N_A, N_B, B\}_{K_{AB}}$$
$$B \rightarrow A: \{N_B, N_A\}_{K_{AB}}$$

协议 4 不是两个协议 2 的组合，消息从 4 个减少到 3 个。

以上 4 个协议中，从 A 发出的加密消息中是否含有 B 的标识是可选的，协议 3 中从 B 发出的消息中是否包含 A 的标识也是可选的。标准建议包含实体的标识用于防御反射(reflection)攻击。

若协议 4 的消息 2 中的标识 B 被省略了，将会遭受如下所示的反射攻击。

1. $B \to C(A): N_B$

1′. $C(A) \to B: N_B$

2′. $B \to C(A): \{N'_B, N_B\}_{K_{AB}}$

2. $C(A) \to B: \{N'_B, N_B\}_{K_{AB}}$

3. $B \to C(A): \{N_B, N'_B\}_{K_{AB}}$

即攻击者 C 参与了与 B 之间的两个并行协议会话，B 是会话 1 的发起者，C 是会话 2 的发起者。首先，C 可以利用 B 发起会话 1 的消息 1 来发起会话 2；接着 C 可以把会话 2 的消息 2′转发给 B 作为会话 1 的消息 2；最后，当会话 1 成功完成后，C 又可以把会话 1 中的消息 3 转发给 B，从而完成会话 2。于是，C 和 B 成功地完成了两次协议运行，而 B 误认为他是和 A 成功地完成了两次协议运行。

思考 10.4　上述协议是否可以通过散列函数完成。

可以通过散列函数完成，将加密函数用具有密钥的散列函数替换即可。唯一的区别是身份信息不再是保密的。例如，协议 1 中 $A \to B: \{T_A, B\}_{K_{AB}}$ 替换为 $A \to B: \text{Hash}_{K_{AB}}(T_A, B), T_A, B$。相应地，其他协议作类似修改。

思考 10.5　如果通信双方事先没有共享密钥，则需要有第三方认证服务器 AS 参与(通常情况下，AS 也承担了密钥分发的任务，此时叫做 KDC)。如何设计认证协议？

对于单向认证：

$A \to AS: A, B, N_A$

$AS \to A: \{K_{AB}, B, N_A, \{K_{AB}, A\}_{K_B}\}_{K_A}$

$A \to B: \{K_{AB}, A\}_{K_B}, M_{K_{AB}}$

对于双向认证：

$A \to AS: A, B, N_A$

$AS \to A: \{K_{AB}, B, N_A, \{K_{AB}, A\}_{K_B}\}_{K_A}$

$A \to B: \{K_{AB}, A\}_{K_B}$

$B \to A: \{N_B\}_{K_{AB}}$

$A \to B: \{f(N_B)\}_{K_{AB}}$

在后来的研究中发现消息 2 和 3 需要加上时间戳。这其实就是 Needham/Schroeder 协议(含有双向认证和共享对称密钥分发)。

另外，Needham/Schroeder 协议在实际中的应用是 Kerberos 协议，它是基于对称密码的实体认证协议，在没有事先共享对称密钥的情况下，由可信第三方参与，同时完成密钥的分发。Kerberos 协议在实际中有大量应用，可以体会实际应用中的协议设计和协议设计原型之间的异同。

10.3.2 基于公钥密码的实体认证

基于公钥密码的实体认证的优势在于：可以利用数字签名提供抗抵赖性，并可以简化密钥的管理，不需要在线的可信第三方。但同时也付出了两个代价：①公钥密码体制中的计算代价都比较高，多需要两三个数量级的计算，因此，在设计基于公钥的协议时，应尽可能地减少公钥操作的数量。进一步将公钥的基本操作细分，需要考虑协议中使用到的私钥操作（如签名生成和解密）和公钥操作（签名验证和加密），这两种操作的效率往往不同。如 RSA 算法的公钥操作比私钥操作的效率要高，而大多数离散对数算法刚好相反。此外，虽然基于离散对数的算法（包括椭圆曲线算法）总体上效率高于 RSA，但对于主要需要使用公钥操作的协议来说，使用小公开指数的 RSA 实现将会更加有效。②需要对公钥进行管理，通常使用公钥基础设施。

本节使用的符号主要有：$E_X(M)$表示用实体 X 的公钥对消息 M 进行加密，$\mathrm{Sig}_X(M)$ 表示用实体 X 的私钥对消息 M 进行有附录的签名，N_X 表示实体 X 选择的随机数，T_X 表示实体 X 选择的时间戳。

ISO/IEC 9798—3 包括 5 个认证协议，美国标准 FIPS196 包括了其中的两个协议（协议 1 和协议 2）。在标准中每个协议的每条消息都包含多个可选的文本域，包含文本域的原因可能是多方面的：如为了认证信息；为了在签名中添加额外的冗余信息；为了提供其他的时间变量参数，如时间戳；为了在协议的使用中提供有效性信息。没有签名的文本域可能用于保存消息发送者的身份标识。由于可选的文本域并不是基本协议的组成部分，故在下面的描述中省略了。另外，如果接收方没有发送方的数字证书，可以在消息中包含发送方的数字证书。

协议 1：一路单向认证协议。

发起者 A 发送一条消息，验证者 B 验证 A 的身份，为单向认证。B 由时间戳 T_A 判断 A 是否有效，包含身份标识 B 能保证 A 知道 B 是期望的验证实体。容易看到和 ISO/IEC 9798—2 一路单向认证非常相似。

$$\mathrm{A} \rightarrow \mathrm{B}: T_\mathrm{A}, \mathrm{B}, \mathrm{Sig}_\mathrm{A}(T_\mathrm{A}, \mathrm{B})$$

协议 2：两路单向认证协议。

与协议 1 相比，该协议用随机数代替时间戳，另外一个不同是包含了 A 选择的随机数 N_A，N_A 与认证无关，但保证了 A 不是对 B 选择的消息进行签名。该协议仍然是单向认证。

$$\mathrm{B} \rightarrow \mathrm{A}: N_\mathrm{B}$$

$$\mathrm{A} \rightarrow \mathrm{B}: N_\mathrm{A}, N_\mathrm{B}, \mathrm{B}, \mathrm{Sig}_\mathrm{A}(N_\mathrm{A}, N_\mathrm{B}, \mathrm{B})$$

协议 3：两路双向认证协议。

该协议是协议 1 的两个实例的简单组合。同样地，时间戳可用计数器代替。这个协议提供双向认证。因为协议中的消息彼此独立，所以协议可以只执行一轮。

$$\mathrm{A} \rightarrow \mathrm{B}: T_\mathrm{A}, \mathrm{B}, \mathrm{Sig}_\mathrm{A}(T_\mathrm{A}, \mathrm{B})$$

$$\mathrm{B} \rightarrow \mathrm{A}: T_\mathrm{B}, \mathrm{A}, \mathrm{Sig}_\mathrm{B}(T_\mathrm{B}, \mathrm{A})$$

协议 4：三路双向认证协议。

该协议是对协议 2 的扩展，A 和 B 分别使用随机数 N_A 和 N_B。

$$B \to A: N_B$$

$$A \to B: N_A, N_B, B, Sig_A(N_A, N_B, B)$$

$$B \to A: N_B, N_A, A, Sig_B(N_B, N_A, A)$$

协议 5：两路并行认证协议。

该协议允许认证在 A 和 B 之间并行进行，因此，消息 1 和 1′，消息 2 和 2′可以同时发送。

1. $A \to B: N_A$

1′. $B \to A: N_B$

2. $A \to B: N_A, N_B, B, Sig_A(N_A, N_B, B)$

2′. $B \to A: N_B, N_A, A, Sig_B(N_B, N_A, A)$

身份认证协议通常和密钥分发或建立协议联合使用，即在认证身份的同时进行密钥的分发或者建立，例如 X. 509 认证协议，Needham-Schroeder 协议（以及该协议的扩展 Kerberos 协议）。

10.3.3　基于散列函数的实体认证

如图 10.8 所示，验证者发送给声称者一个自己产生的随机挑战 n，声称者输入口令 p'，p'和 id 经过散列函数 f 计算的结果（等于 q）再和 n 通过散列函数 h 计算得到 r'，r'发送给验证者。验证者将保存的 q 和生成的 n 通过散列函数 h 计算得到 r，通过比较 r 和 r' 是否相等来确定认证是否通过。

该协议中非重复的挑战完全由验证者决定，使得每次传输的认证信息不同，很好地防止了口令窃听和重放，但需要额外的通信开销。其安全性一方面取决于散列函数的安全性；另一方面由于是单向认证，存在验证者假冒和重放攻击。这可以通过双向认证或时间戳来解决。

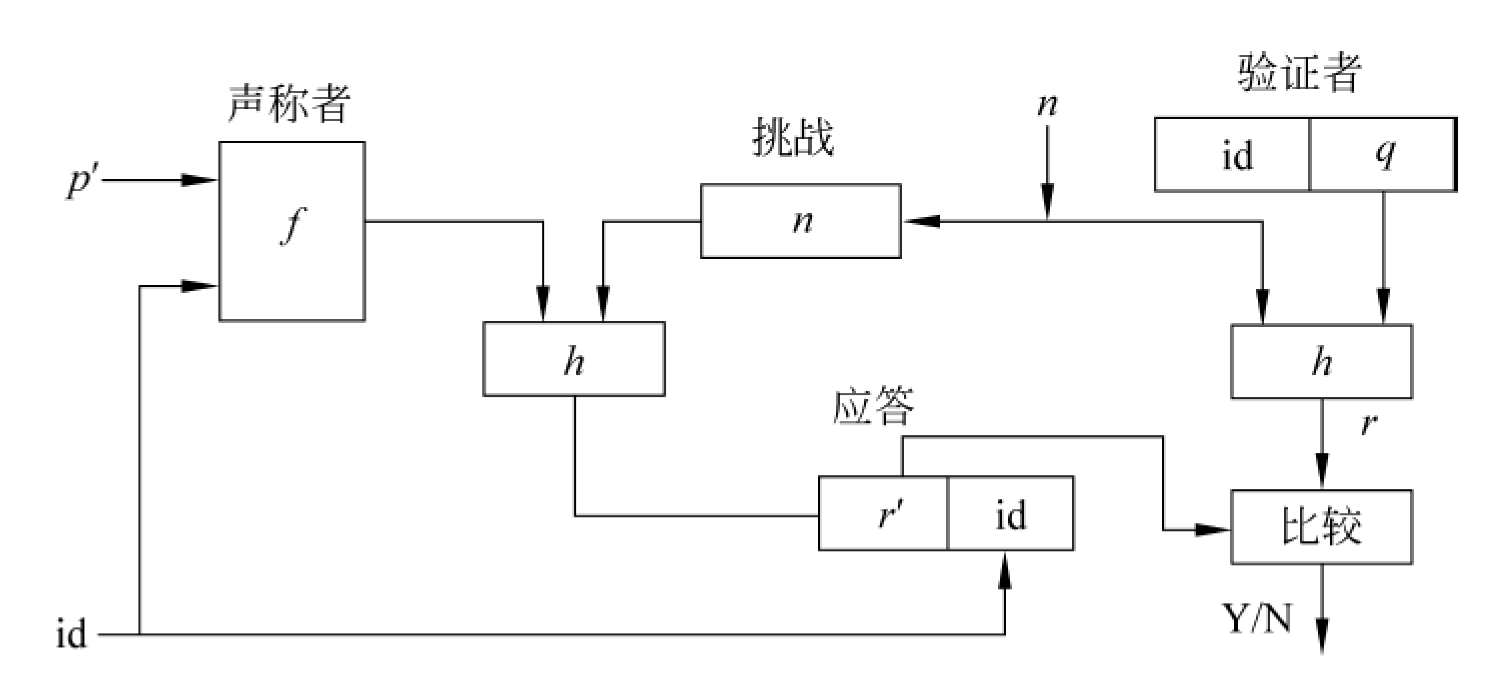


图 10.8　基于散列函数的实体认证示意图

第 11 章

身份识别协议

身份识别协议使用非对称技术证明身份,但不依赖于数字签名或公钥加密,而且可避免使用分组密码、序列号和时间戳。形式上类似于基于口令的实体认证,拥有“挑战-应答”的交互过程,但是这里的协议是基于交互式证明和零知识证明的思想。不但利用随机数作为挑战,也利用随机数作为阻止欺骗的一个承诺。身份识别协议的设计原理要在零知识证明协议学习完之后才可以完全理解,本章重点是介绍身份识别协议。不同于前面的实体认证协议,身份识别协议可证明自己的身份,但验证者不能假冒证明者。

另外,前面曾经提到,所有身份识别协议可以通过利用一个散列函数将交互式证明变为非交互式证明,从而得到相应的签名方案。

11.1 Fiat-Shamir 身份识别协议

首先介绍交互证明系统的概念。

交互证明系统由两方参与,分别称为证明者(Prover,简记为 P)和验证者(Verifier,简记为 V),其中 P 知道某一秘密(如公钥密码体制的私钥),P 希望能使 V 相信自己的确掌握这一秘密。交互证明由若干轮组成,在每一轮,P 和 V 可能需根据从对方收到的消息和自己计算的某个结果向对方发送消息。比较典型的方式是在每一轮 V 都向 P 发出询问(挑战),P 向 V 做出应答。所有轮执行完后,V 根据 P 是否在每一轮对自己发出的询问都能正确回答来决定是否接受 P 的证明。交互证明有时称为协议证明。

交互证明和数学证明的区别在于:数学证明的证明者可独自完成证明,而交互证明是由 P 产生证明、V 验证证明的有效性来实现,因此是双方之间通过某种信道的通信时必需的。

交互证明系统需要满足以下要求:

(1) 完备性(completeness)。给定 P、V 都是诚实的,如果 P 知道某一秘密,V 将以很高的概率接受 P 的证明。

(2) 合理性(soundness)。如果 P 能以不可忽略的概率使 V 相信 P 的证明,则 P 知道相应的秘密。

交互证明如果具有完备性和合理性,则称为知识证明。知识的证明协议有零知识性。即证明者执行协议(即使和恶意验证者交互)不会泄露任何在多项式时间可公开计算的信息之外的信息,因此,验证者不能获得关于 P 的有用信息,使得 V 不能随后假冒 P 向第三方证明自己是 P。

更正式地说，零知识性就是在下述意义下是可模拟的：存在多项式算法（模拟器），在未与真正的证明者交互的情形下，一旦输入要证实的断言，该算法就能生成一些脚本，这些脚本和那些与真正的证明者交互而得来的脚本不可区分。

1986 年，A. Fiat 和 A. Shamir 在 Crypto'86 上提出了一个基于模 n 平方根问题的困难性的身份识别协议，其安全性等价于因子分解 n。协议的设计运用了"挑战-应答"机制，以及"分割-选择"思想，并保证了零知识性。

Fiat-Shamir(FS)身份识别协议的目的是 A 向 B 证明对知识 s 的拥有，执行 t 次协议。

Fiat-Shamir 身份识别协议描述如下：

1. 参数设置：
   可信中心 TA 选择并公开模数 $n=pq$，素数 p 和 q 保密。
   每个声称者 A 选择与 n 互素的秘密 s，$1\leqslant s\leqslant n-1$，计算 $v=s^2 \bmod n$，将 v 向 TA 注册为自己的公钥。
2. 协议消息：t 轮交互的每一轮都由如下 3 条消息构成：

$$A\rightarrow B: x=r^2 \bmod n$$
$$A\leftarrow B: e\in\{0,1\}$$
$$A\rightarrow B: y=r\cdot s^e \bmod n$$

3. 协议执行过程：下列步骤连续地执行 t 次，假如 t 轮都成功，B 就是接受证明。
   (1) A 选择随机数（承诺）r，$1\leqslant r\leqslant n-1$，发送（证据）$x=r^2 \bmod n$ 给 B。
   (2) B 随机选择一个（挑战）比特 $e=0$ 或 $e=1$，发送 e 给 A。
   (3) A 计算（响应）y，并发送给 B，其中 $y=r(e=0)$，或者 $y=rs \bmod n(e=1)$。
   (4) 若 $y=0$，则 B 拒绝证明，反之则通过验证 $y^2=x\cdot v^e \bmod n$ 接受证明（即若 $e=1$，则 $y^2\equiv xv \bmod n$；若 $e=0$，则 $y^2\equiv x \bmod n$）。

该协议满足以下要求：

(1) 完备性。如果 P 和 V 遵守协议，且 P 知道 s，则应答 rs 应为模 n 下 xv 的平方根，V 接受 P 的证明。

(2) 合理性。如果 P 不知道 s，他在第 1 条消息中发送 $x=r^2/v$，在第 3 条消息中发送 $y=r$ 作为应答。当挑战为 $e=1$，这个应答是正确的，即满足方程 $[M_3]^2=[M_1]v \bmod n$，这里 $[M_3]$ 表示第 3 条消息，$[M_1]$ 表示第 1 条消息。当挑战为 $e=0$ 时，则应答不正确。故可欺骗的概率为 1/2。如果进行 t 轮均正确应答，则欺骗的概率为 2^{-t}。

(3) 零知识性。V 并没有知道 P 的秘密，P 给 V 的消息中只能得出 P 知道 v 的平方根这一事实。

思考 11.1　基于零知识的身份识别协议的设计规律。

一般来说，这类协议的一般结构是

A → B：证据（承诺）
A ← B：挑战
A → B：响应

需要证明自己是 A 的证明者从预定义集中选择一个随机元素作为秘密承诺，并由此计算相关的公开证据。这提供了协议运行的随机初始化，实质上定义了证明者声称能够

回答的一套问题，限制了随机后的响应。后面的协议中，只有知道秘密的合法方A才有能力回答所有的挑战，对这些挑战的响应不会泄露任何A的秘密信息。B在随后的挑战中选择其中一个问题，A提供响应给B，B检测响应的正确性。必要时，重复执行协议，以降低成功欺骗概率的上界。

设计中应用到“挑战-应答”的形式，利用了“切割-选择”协议的思想（源于两个小孩分一块蛋糕的方法：一个切，另一个选）。对一个给定的证据，A最多响应一个挑战，而且不重复使用任何证据。如果违背这一条件，安全性就会受到威胁。

11.2 Feige-Fiat-Shamir 身份识别协议

为了提高FS身份识别协议的效率，1988年，U. Feige、A. Fiat和A. Shamir提出了Feige-Fiat-Shamir(FFS)身份识别协议，该协议是FS身份识别协议的推广。

Feige-Fiat-Shamir身份识别协议描述如下：

1. 参数设置：可信中心TA选择两个秘密素数 p 和 q，使得 $n=pq$ 的因子分解在计算上是不可行的，其中 p 和 q 都为模4余3，将 n 作为公共的模数向所有的用户公开（n 是Blum数）。整数 k 和 t 定义为安全参数。
   每个实体选择 k 个随机整数 $s_1, s_2, \cdots, s_k (1 \leqslant s_i \leqslant n-1)$ 和 k 个随机比特 $b_1, b_2, \cdots, b_k$。
   计算 $v_i = (-1)^{b_i} \cdot (s_i^2)^{-1} \bmod n, 1 \leqslant i \leqslant k$。A向TA证明自己的身份，TA注册A的公钥 $(v_1, v_2, \cdots, v_k; n)$，只有A知道自己的私钥 $(s_1, s_2, \cdots, s_k)$ 和 n。
2. 协议消息。t 轮中的每轮执行如下3条消息：

$$A \rightarrow B: x = \pm r^2 \bmod n$$

$$A \leftarrow B: (e_1, e_2, \cdots, e_k), e_i \in \{0,1\}$$

$$A \rightarrow B: y = r \cdot \prod\nolimits_{e_j=1} s_j \bmod n$$

3. 协议执行过程。下列步骤执行 t 次，假如 t 轮都成功，B就接受A的身份。假定B有A的可信公钥 $(v_1, v_2, \cdots, v_k; n)$，否则，可在第1条消息中发送证书。
   (1) A选择随机整数 $r(1 \leqslant r \leqslant n-1)$ 和一个随机比特 b；计算 $x = (-1)^b \cdot r^2 \bmod n$；发送 x（证据）给B。
   (2) B将随机 k 比特向量 $(e_1, e_2, \cdots, e_k)$ 发送给A（挑战）。
   (3) A计算并发送给B（响应）：$y = r \cdot \prod_{j=1}^{k} s_j^{e_j} \bmod n$（根据挑战得出这些 s_j 和 r 的乘积）。
   (4) B计算 $z = y^2 \prod_{j=1}^{k} v_j^{e_j} \bmod n$，验证 $z = \pm x$ 和 $z \neq 0$（后者是排除敌手选择 $r=0$ 能够成功的情形）。

可见，FFS方案和FS方案的区别在于生成证据和挑战都是多个的（另外一个区别是验证方程有点改动，这是因为公钥和私钥方程有改动）。因此，某种意义上来说，FFS是并行的FS方案。

例 11.1 TA取 $p=7, q=13$，计算Blum数 $n=7\times13=91$ 并公布。不妨设P的私钥是 $s_1=9, s_2=31, s_3=67$，公钥是 v_1, v_2, v_3，由 $s_1^2 v_1 = 1 \bmod 91$，得到 $v_1=9$。同法可得到 $y_2=25, y_3=88$。

协议执行如下：

(1) P 随机选择一个整数 $r=57$，计算 $x=57^2 \bmod 91=64$，把 $x=64$ 发送给 V。

(2) 取 $e=\{1,1\}$，发送 e 给 P。

(3) P 计算 $y=r\cdot\prod_{j=1}^{2}s_j^{e_j} \bmod n=57s_1s_2 \bmod n=57\times9\times31 \bmod 91=69$，发送给 V。

(4) V 计算 $z=y^2\prod_{j=1}^{2}v_j^{e_j}=69^2v_1v_2 \bmod 91=69^2\times9\times25 \bmod 91=64$，验证通过。

(5) 重复(1)～(4)t 次。

思考 11.2　FFS 方案的假冒成功的概率是多少。

每一轮假冒成功的概率为 $1/2^k$，t 轮之后假冒成功的概率为 2^{-kt}。

(对照：FS 方案每一轮假冒成功的概率为 1/2，t 轮之后假冒成功的概率为 2^{-t}。)

11.3　Guillou-Quisquater 身份识别协议

前面两个方案均基于模 n 的平方根问题的困难性。下面介绍一种基于大整数分解问题的身份识别协议 Guillou-Quisquater(GQ)方案，是由 L. C. Guillou 和 J-J. Quisquater 给出的。

GQ 身份识别协议描述如下：

1. 参数设置。

该方案需要一个可信机构 TA。TA 首先选取两个大素数 p 和 q，计算 $n=pq$，p 和 q 保密，n 公开。TA 选择一个大素数 b 作为公开参数，也称为公共的 RSA 加密指数。通常 b 是满足 $\gcd(b,\phi(n))=1$ 的 40 比特长的素数。

证明者 A 选择一个整数 u，满足 $0\leqslant u\leqslant n-1$，计算 $v=(u^{-1})^b \bmod n$，并传递给 TA，TA 计算签名 $\mathrm{Sign}_{TA}(\mathrm{ID}_A \parallel v)$，证书 $\mathrm{Cert}_A=\{\mathrm{ID}_A, v, \mathrm{Sign}_{TA}(\mathrm{ID}_A \parallel v)\}$。整数 n 和 v 是公开参数，v 是 A 的公钥，u 是 A 的私钥。

2. 协议消息。

$$A \to B:\ \mathrm{Cert}_A, x = r^b \bmod n$$
$$A \leftarrow B:\ e(1 \leqslant e \leqslant b-1)$$
$$A \to B:\ y = ru^e \bmod n$$

3. 协议执行。

(1) A 选择随机数 r，$0\leqslant r\leqslant n-1$，计算 $x=r^b \bmod n$。

(2) A 传送 Cert_A 和 x 给 B。

(3) B 验证 Cert_A。

(4) B 选择随机数 e，$0\leqslant e\leqslant b-1$，并传送 e 给 A。

(5) A 计算 $y=ru^e \bmod n$，并传送响应 y 给 B。

(6) B 验证是否有 $x\equiv v^e y^b \bmod n$。

该协议满足以下要求：

(1) 完备性。A 如果确实知道其秘密信息(私钥)u，则可以对任意挑战计算响应。易知 $v^e\cdot y^b \bmod n=v^e(ru^e)^b \bmod n=(vu^b)^e r^b \bmod n=r^b \bmod n=x$。

(2) 合理性。如果伪装成 A 的攻击者能猜出 B 的挑战 e，则在提交 x 时先选定 y，并取 $x=v^e y^b \bmod n$，其后在响应消息中发送选定的 y，则会导致满足验证方程。这种假冒

成功的概率即为猜中 e 的概率，即约为 $1/b$。

11.4 Schnorr 身份识别协议

C. P. Schnorr 在 1989 年设计了一种身份识别协议，该协议基于离散对数问题的困难性。该协议的计算量和通信量都不大，特别适合计算机能力有限的终端设备。

在介绍 Schnorr 身份识别协议之前，先总结一下 FS 和 GQ 两种方案的共同点(见表 11.1)，从而体会如何基于离散对数问题设计身份识别协议。

表 11.1 FS 和 GQ 身份识别协议的对比

协议	参数设置	协议消息	协议验证
FS	证明者私钥：s 证明者公钥： $v=s^2 \bmod n$	A→B：$x=r^2 \bmod n$ A←B：$e\in\{0,1\}$ A→B：$y=rs^e \bmod n$	$y^2\equiv xv^e \bmod n$
GQ	证明者私钥：u 证明者公钥： $v=(u^{-1})^b \bmod n$	A→B：$\text{Cert}_A, x=r^b \bmod n$ A←B：$e(1\leqslant e\leqslant b-1)$ A→B：$y=ru^e \bmod n$	$x\equiv v^e y^b \bmod n$

从表 11.1 中可以发现一个一般规律：第一条消息表明：根据协议所基于的困难问题，提交对一个随机值 r 的承诺 commit。第二条消息：返回挑战 challenge。第三条消息：根据私钥 s(或 u)返回响应，形如 $\text{response}=r\cdot s^{\text{challenge}}$。对此的理解在承诺协议介绍后会更加清楚。

根据上述一般规律，下面介绍 Schnorr 身份识别协议(以下改变了描述方式，以表述一般规律)。A 与 B 分别是证明者和验证者。

1. 参数设置。

   两个大素数 p 和 q，满足 $q|(p-1)$。$\mathbb{Z}_p$中的阶为 q 的元素 p。v 是$\mathbb{Z}_p$中的元素，这里 $v=g^{-s} \bmod p, s\in\mathbb{Z}_q$。

   于是(p,q,g,v)是经过 TA 证实的 A 的公钥，A 的私钥是 s。

2. 协议执行 $t=\log_2\log_2 p$ 次。

   (1) A 选择 $r\in\mathbb{Z}_q$，计算 $\text{commit}=g^r \bmod p$，发送给 B。

   (2) B 选择 challenge($1\leqslant \text{challenge}\leqslant 2^t<q$)，发送给 A。

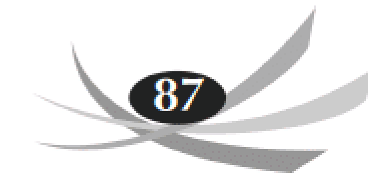
   (3) A 置 $\text{response}=r+s\cdot \text{challenge} \bmod q$，发送给 B。

   (4) B 验证等式 $\text{commit}=g^{\text{response}}v^{\text{challenge}} \bmod p$ 是否成立。

Schnorr 身份识别协议的优点是可以预先计算指数运算(承诺时)，声称者只需要在线计算(应答时)一次模乘法。响应时的计算量比 FS 方案和 GQ 方案小。但验证者的计算量较大。

例 11.2 选择素数 $p=48731$，其中 $p-1$ 能够被 $q=443$ 整除。模 48731 的生成元是 6，计算 $g=6^{(p-1)/q} \bmod p=11444$。系统参数为(48731,443,11444)。选择参数 $t=8$。

A 选择一个私钥 $s=357$，计算 $v=g^{-s} \bmod p=11444^{-357} \bmod 48731=7355$。



协议消息交换过程如下：

(1) A 选择 $r=274$，发送 $x=g^r \bmod p=11444^{274} \bmod 48731=31123$ 给 B。

(2) B 发送给 A 随机挑战 challenge=129。

(3) A 发送给 B 响应：

$$\text{response}=r+s\cdot\text{challenge} \bmod q=274+357\times129 \bmod 443=255$$

(4) B 验证

$$g^{\text{response}}v^{\text{challenge}} \bmod p=11444^{255}\times7355^{129} \bmod 48731=31123=x$$

11.5　Okamoto 身份识别协议

目前还没有发现 Schnorr 身份识别协议是可证明安全的。在 1992 年的 Crypto'92 会议上，T. Okamoto 提出了 Schnorr 身份识别协议的改进版，具有可证明安全性。

Okamoto 身份识别协议描述如下：

1. 参数设置。

   TA 选择两个大素数 p 与 q，满足 $q\mid(p-1)$。$\mathbb{Z}_p$ 中的阶为 q 的元素。A 秘密选择 $s_1,s_2\in\mathbb{Z}_q$，并计算出 $\mathbb{Z}_p$ 中元素 $v=g_1^{-s_1}g_2^{-s_2} \bmod p$。$(p,q,g_1,g_2,v)$ 是 TA 证实的 A 的公钥。

2. 协议消息：

   (1) A 选择 $r_1,r_2\in\mathbb{Z}_q$，计算 $\text{commit}=g_1^{r_1}g_2^{r_2} \bmod p$，发送给 B。

   (2) B 选择 challenge($1\leqslant\text{challenge}\leqslant2^t<q$) 发送给 A。

   (3) A 置

$$\text{response}_1=r_1+s_1\cdot\text{challenge} \bmod q$$
$$\text{response}_2=r_2+s_2\cdot\text{challenge} \bmod q$$

   把 response_1，response_2 发送给 B。

   (4) B 验证 $\text{commit}=g_1^{\text{response}_1}g_2^{\text{response}_2}v^{\text{challenge}} \bmod p$ 是否成立。

可见，Okamoto 方案和 Schnorr 方案的区别在于前者使用了两个生成元 g_1,g_2。由于当 p 与 q 较大时，计算离散对数问题 $\log_{g_1}g_2$ 是困难的，这也正是 Okamoto 方案具有可证明安全性的原因。但是，Okamoto 方案比 Schnorr 方案的计算量大，因而速度与效率较低。

最后需要指出的是：身份识别协议可以转换为签名方案，方法是证明者自己构造一个挑战，这一挑战的构造方法通常是使用单向散列函数，以达到对挑战值的不可预测性。具体而言，使用一个公开的安全散列函数来代替验证者提出挑战值，即 $\text{challenge}=H(m,\text{commit})$。这种方法称为 Fait-Shamir 启发式方法。例如，对 m 的签名为 (commit, response)，其中 $\text{commit}=g^r \bmod p$，$\text{response}=(r+sH(x,m)) \bmod q$（$s$ 为私钥）。签名验证方程为 $\text{commit}=g^{\text{response}}v^{H(x,m)} \bmod p$（$v$ 为公钥）。

第 12 章 密钥协商协议

公钥加密优于对称密钥加密的地方在于公钥加密不需要安全信道传送密钥，但是大多数公钥密码体制的加密和解密速度比对称密钥慢得多，因而公钥密码通常只适合对少量数据的加密，而大量或者频繁的数据加密仍然需要依靠对称密钥加密。通信中通常需要建立网络用户间的共享(会话)密钥，如果在线可信当局(trust authority，TA)不适用或者不希望有一个在线 TA，则可以利用密钥协商(key agreement)协议来建立通信双方之间的共享密钥。利用密钥协商协议建立的共享密钥是通信双方输入消息的一个函数。

12.1 两方密钥协商

12.1.1 Diffie-Hellman 密钥协商协议

W. Diffie 和 M. Hellman 在 1976 年提出 Diffie-Hellman 密钥协商(key negotiation)协议，目的在于使通信双方利用该协议可以在一个公开的信道上建立共享的会话密钥。该协议受到了 Merkle 谜题协议设计思想的影响，该协议的别名还包括 Diffie-Hellman 密钥交换(key exchange)、Diffie-Hellman 密钥认同(key agreement)、Diffie-Hellman 密钥建立(key establishment)、指数密钥交换(exponential key exchange)、Diffie-Hellman 协议(protocol)和 Diffie-Hellman 握手(handshake)。据说在此之前为英国智囊机构——政府通信指挥部(Government Communications Headquarters，GCHQ)工作的(Malcolm Williamson)就已经发明了该方法，但 GCHQ 直到 1997 年才公布，因而在学术界没什么影响。

Diffie-Hellman 密钥协商协议描述如下(图 12.1)。

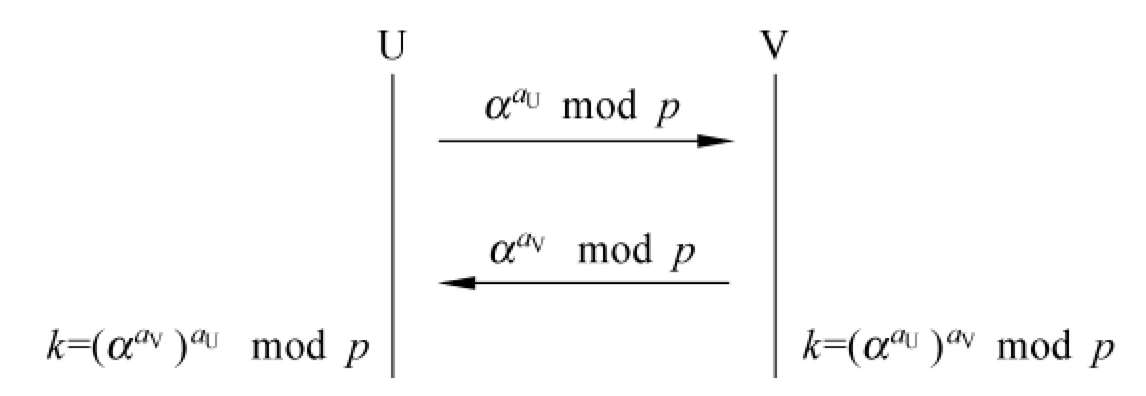


图 12.1 Diffie-Hellman 密钥协商协议

设 p 是一个大素数，$\alpha\in\mathbb{Z}_p$ 是一个本原元。p 和 α 公开(也可由 U 发送给 V)。

(1) 用户 U 随机选取 a_U，$2\leqslant a_U\leqslant p-2$。

(2) 用户 U 计算 α^{a_U} mod p，并将结果传送给用户 V。

(3) 用户 V 随机选取 a_V，$2 \leqslant a_V \leqslant p-2$。

(4) 用户 V 计算 $\alpha^{a_V} \bmod p$，并将结果传送给用户 U。

(5) 用户 U 计算 $k=(\alpha^{a_V})^{a_U} \bmod p$。

(6) 用户 V 计算 $k=(\alpha^{a_U})^{a_V} \bmod p$。

于是，k 为用户 U 和用户 V 的共享会话密钥。

不难看出，Diffie-Hellman 密钥协商协议的安全性主要是基于有限域上离散对数问题的难解性。

例 12.1　设 $p=27803$，$\alpha=5$。p 是素数，$\alpha \in \mathbb{Z}_p$是一个本原元。

假设用户 U 选取 $a_U=169$，计算 $b_U=\alpha^{a_U} \bmod p=5^{169} \bmod 27803=6268$，然后用户 U 将结果 $b_U=6268$ 传送给用户 V。

假设用户 V 选取 $a_V=23456$，计算 $b_V=\alpha^{a_V} \bmod p=5^{23456} \bmod 27803=26759$，然后用户 V 将结果 $b_V=26759$ 传送给用户 U。

当用户 U 收到 b_V 后，计算密钥 $k=b_V^{a_U} \bmod p=26759^{169} \bmod 27803=6998$。

当用户 V 收到 b_U 后，计算密钥 $k=b_U^{a_V} \bmod p=6268^{23456} \bmod 27803=6998$。

用户 U 和用户 V 之间的共享密钥为 $k=6998$。

Diffie-Hellman 密钥协商协议的原理如下。

通信双方统一两个数 D 和 H。D 与 H 无须对外保密。双方各自选择一个秘密的数 X，并把包括 D，H 和 X 的计算结果 Y 发送给对方。例如，假设甲方选中了 X_1，则发出了 Y_1；乙方选中了 X_2，则发出了 Y_2。

根据算法，双方各自使用自己选择的 X 和收到的 Y，计算出一个统一的数字 K。K 就是双方进一步通信的会话密钥。具体地讲，K 可以从 (X_1, Y_2) 或 (X_2, Y_1) 中导出，但不能从 (Y_1, Y_2) 中得到。这一点的重要意义在于，窃听者虽然可以得到 D，H，Y_1 和 Y_2，甚至密文，但由于不知道 X_1 和 X_2 的值，因此无法导出正确的会话密钥 K，从而也就无法破解密文。

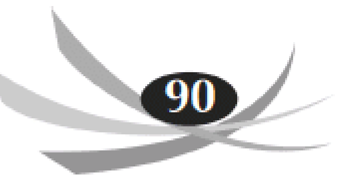
以后通信时，双方便使用 K 进行加密和解密。

从这个一般性讨论中可以发现，Diffie-Hellman 算法的局限在于它需要收发双方必须同时参与密码的生成过程，所以它不适合电子邮件的加密，因为电子邮件在收信人不在场时也应当能够发送过去。

Diffie-Hellman 密钥协商协议可以很容易地推广到椭圆曲线上。椭圆曲线上的 Diffie-Hellman 密钥协商协议描述如下(图 12.2)。

图 12.2　Diffie-Hellman 密钥协商协议的椭圆曲线版本

设 E 是有限域$\mathbb{Z}_p$上的椭圆曲线，$P \in E$，并且 P 的阶足够大，使得由 P 生成的循环群



上的离散对数问题是难解的。设 n 是点 P 的阶。E 和 P 公开。

(1) 用户 U 随机选取整数 n_U, $1 \leqslant n_U \leqslant n$。

(2) 用户 U 计算 $n_U P \bmod p$,并将结果传送给用户 V。

(3) 用户 V 随机选取整数 n_V, $1 \leqslant n_V \leqslant n$。

(4) 用户 V 计算 $n_V P \bmod p$,并将结果传送给用户 U。

(5) 用户 U 计算 $k = n_U(n_V P) \bmod p$。

(6) 用户 V 计算 $k = n_V(n_U P) \bmod p$。

k 为用户 U 和用户 V 的共享会话密钥。

不难看出,椭圆曲线上的 Diffie-Hellman 密钥协商协议的安全性主要是基于椭圆曲线上离散对数问题的难解性。

Diffie-Hellman 密钥协商协议非常简洁,许多密钥协商协议都是在其基础上设计而成的。Diffie-Hellman 密钥协商协议在安全性能上看起来没有什么问题,但其实它容易受到主动敌手的中间人攻击(Man-in-the-Middle Attack, MITM Attack)。

假设 A 是一个主动敌手。A 对 Diffie-Hellman 密钥协商协议的中间人攻击如图 12.3 所示。

图 12.3 对 Diffie-Hellman 密钥协商协议的中间人攻击

A 在对 Diffie-Hellman 密钥协商协议进行中间人攻击时,插在用户 U 和用户 V 之间,截获 U 和 V 传送的消息并替换成自己的消息。在协议结束时,用户 U 实际上与 A 建立了一个共享密钥 $\alpha^{a'_V a_U}$,用户 V 也与 A 建立了一个共享密钥 $\alpha^{a'_U a_V}$,而用户 U 和用户 V 之间并没有建立一个共享的密钥。在这种情况下,当用户 U 将消息加密传送给用户 V 时,A 能解密从而获得消息,而 V 不能解密从而无法获得消息。反过来,当用户 V 向用户 U 传送加密消息时也存在同样的问题。

思考 12.1 如何抵抗针对 Diffie-Hellman 密钥协商协议的中间人攻击?

抵抗中间人攻击的方法是当用户 U 和用户 V 接收到对方传送过来的消息时进行身份认证。

12.1.2 端到端密钥协商协议

W. Diffie 和 P. C. Van Oorschot 以及 M. J. Wiener 于 1992 年给出的端到端(Station-to-Station, STS)协议是对 Diffie-Hellman 密钥协商协议的一个修改。端到端协议可以抵抗中间人攻击。在端到端协议中,假设每个用户 U 都可以签名,签名生成算法为 Sign_U,签名验证算法为 Vrfy_U。假设 TA 也有一个签名方案,签名算法为 Sign_{TA},公开的签名验证算法为 Vrfy_{TA}。TA 给每个用户 U 事先发放一个证书(certificate):

$$C(U) = (\mathrm{ID}(U), \mathrm{Vrfy}_U, \mathrm{Sign}_{TA}(\mathrm{ID}(U), \mathrm{Vrfy}_U))$$

其中 ID(U)是用户 U 的身份信息。

下面给出简化了的端到端协议。假设 p 是一个大素数,$\alpha \in \mathbb{Z}_p$ 是一个本原元。p 和 α 公开。

(1) 用户 U 随机选取 a_U，$1 \leqslant a_U \leqslant p-2$。

(2) 用户 U 计算 $\alpha^{a_U} \bmod p$，并将结果传送给用户 V。

(3) 用户 V 随机选取 a_V，$1 \leqslant a_V \leqslant p-2$。

(4) 用户 V 首先计算 $\alpha^{a_V} \bmod p$。然后 V 计算 $k=(\alpha^{a_U})^{a_V} \bmod p$，$y_V=\mathrm{Sign}_V(\alpha^{a_V} \bmod p, \alpha^{a_U} \bmod p)$。

(5) 用户 V 将 $(C(V), \alpha^{a_V} \bmod p, y_V)$ 传送给 U。

(6) 用户 U 计算 $k=(\alpha^{a_V})^{a_U} \bmod p$。用户 U 利用 Vrfy_V 验证 y_V，利用 Vrfy_{TA} 验证 $C(V)$。

(7) 用户 U 计算 $y_U=\mathrm{Sign}_U(\alpha^{a_U} \bmod p, \alpha^{a_V} \bmod p)$。然后用户 U 将 $(C(U) y_U)$ 传送给用户 V。

(8) 用户 V 利用 Vrfy_U 验证 y_U，利用 Vrfy_{TA} 验证 $C(U)$。

在端到端协议中，用户 U 和用户 V 之间在信道上传送的消息可以用图 12.4 来说明。

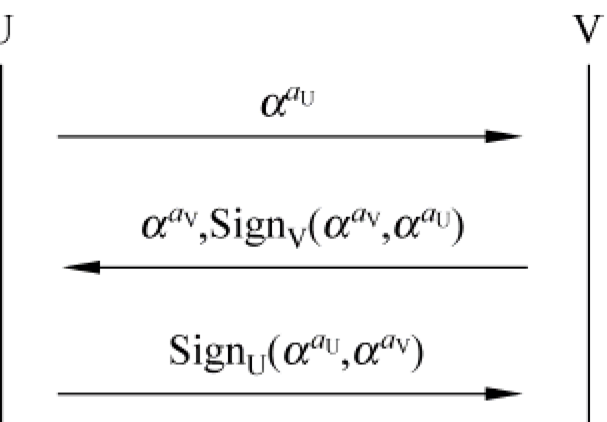
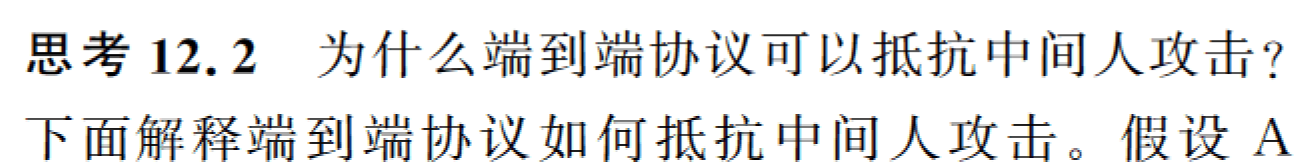


图 12.4 端到端协议中消息的传输



思考 12.2 为什么端到端协议可以抵抗中间人攻击?

下面解释端到端协议如何抵抗中间人攻击。假设 A 是一个主动的敌手，插在用户 U 和用户 V 之间。A 对端到端协议的中间人攻击如图 12.5 所示。A 截获用户 U 发送的 α^{a_U}，将其替换为 $\alpha^{a'_U}$，然后 A 截获用户 V 发送的 α^{a_V} 和 $\mathrm{Sign}_V(\alpha^{a_V}, \alpha^{a'_U})$。A 可以将 α^{a_V} 替换为 $\alpha^{a'_V}$，同时 A 必须将 $\mathrm{Sign}_V(\alpha^{a_V}, \alpha^{a'_U})$ 替换为 $\mathrm{Sign}_V(\alpha^{a'_V}, \alpha^{a_U})$。但是，因为 A 不知道用户 V 的签名算法 Sign_V，所以他无法计算用户 V 对 $(\alpha^{a'_V}, \alpha^{a_U})$ 的签名 $\mathrm{Sign}_V(\alpha^{a'_V}, \alpha^{a_U})$。同样地，由于 A 不知道用户 U 的签名算法 Sign_U，所以也无法将 $\mathrm{Sign}_U(\alpha^{a_U}, \alpha^{a'_V})$ 替换为 $\mathrm{Sign}_U(\alpha^{a'_U}, \alpha^{a_V})$。因此，端到端协议可以抵抗中间人攻击。

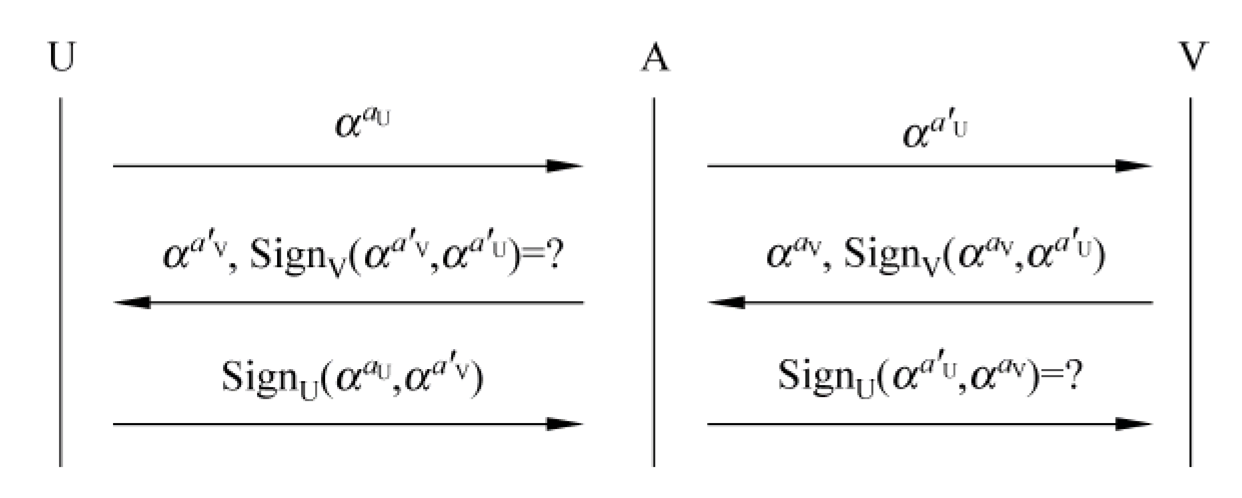


图 12.5 端到端协议对中间人攻击的防御

12.1.3 MTI 密钥协商协议

同端到端密钥协商协议一样，MTI 密钥协商协议是 Matsumoto、Takashima 和 Imai 通过改进 Diffie-Hellman 密钥协商协议而构造的。与端到端相比，它的好处是不需要计算任何签名，另外，它需要从用户 U 到 V 以及从用户 V 到 U 的两次消息传输，而端到端需要三次消息传输。

假设 p 是一个大素数，且 $\alpha(0<\alpha<p)$ 是循环群 $\mathbb{Z}_p$ 的生成元，α 和 p 公开。参与系统的每一个用户(例如 U)都有一个 ID(U)(标识用户的身份信息)，一个私钥 x_U($1 \leqslant x_U \leqslant$

$p-2$）和一个公钥 $y_U=\alpha^{x_U} \bmod p$；TA 有一个数字签名方案，假设其公开的验证算法为 Vrfy_{TA} 和秘密的签名算法 Sign_{TA}；每个用户（例如 U）都有一个证书 $\mathrm{Cert}_U=(\mathrm{ID}(U), y_U, \mathrm{Sign}_{TA}(\mathrm{ID}(U), y_U))$ 绑定其公钥和身份标识。

MTI 密钥协商过程如下：

（1）用户 U 选取一个随机数 $r_U(1\leqslant r_U\leqslant p-2)$，计算 $s_U=\alpha^{r_U} \bmod p$，把 (Cert_U, s_U) 发送给 V。

（2）用户 V 选取一个随机数 $r_V(1\leqslant r_V\leqslant p-2)$，计算 $s_V=\alpha^{r_V} \bmod p$，把 (Cert_V, s_V) 发送给 U。

（3）用户 U 接收到 s_V 后，计算 $K=s_V^{x_U} y_V^{r_U} \bmod p$（$y_V$ 是用户 U 从所接收到的用户 V 的证书 Cert_V 得到的），同时用户 V 计算 $K=s_U^{x_V} y_U^{r_V}$（y_U 是用户 V 从所接收到的用户 U 的证书 Cert_U 得到的）。

由于 $s_V^{x_U} y_V^{r_U}=\alpha^{r_V x_U+x_V r_U}=s_U^{x_V} y_U^{r_V}$，经过两次消息传递后，用户 U 和 V 建立了会话密钥 K。协议执行示意图如图 12.6 所示。

U V
$\mathrm{Cert}_U, s_U=\alpha^{r_U} \bmod p$ →
← $\mathrm{Cert}_V, s_V=\alpha^{r_V} \bmod p$

图 12.6 MTI 协议的交互过程

例 12.2 假设素数 $p=29$，选取有限域 $\mathbb{Z}_{29}$ 的生成元 $\alpha=11$。U 选择私钥 $x_U=15$，得到公钥 $y_U=11^{15} \bmod 29=18$，用户 U 将 y_U 传递给 TA，TA 签名后得到 U 的证书 Cert_U。同样，用户 V 选择私钥 $x_V=23$，公钥为 $y_V=11^{23} \bmod 29=27$。V 将自己的公钥发送给 TA，TA 签名后得到 V 的证书 Cert_V 并发送给 V。用户 U 和 V 运行 MTI 协议建立会话密钥时，用户 U 选择了 $r_U=13$，然后 U 计算 $s_U=11^{13} \bmod 29=21$，把 $(\mathrm{Cert}_U, 21)$ 发送给 V，用户 V 选择 $r_V=25$，计算 $s_V=11^{25} \bmod 29=19$，并把 $(\mathrm{Cert}_V, 19)$ 发送给 U。

现在，用户 U 可计算共享密钥 $K=s_V^{x_U} y_V^{r_U}=19^{15}\times 27^{13} \bmod 29=5$；同样，V 可计算共享密钥 $K=s_U^{x_V} y_U^{r_V}=21^{23}\times 18^{25} \bmod 29=5$。U 和 V 得到相同的密钥。

思考 12.3 MTI 协议为何可以抵抗中间人攻击？

如果出现中间人攻击，则用户 U 和 V 将得到不同的密钥：用户 U 计算得到密钥 $K=\alpha^{r_U x_V+r'_V x_U} \bmod p$，用户 V 计算 $K'=\alpha^{r'_U x_V+r_V x_U} \bmod p$，显然 $K\neq K'$。攻击者无法计算得到任何密钥，因为密钥的运算需要知道私钥 x_U, x_V。即使 U 和 V 计算了不同的密钥，敌手也无法知道这两个密钥中的任何一个。

也就是说，用户 U 和 V 深信另一用户是网络中能计算出自己已计算的密钥的唯一用户，这种特性也称为隐式密钥认证。

12.1.4 ECMQV 密钥协商体制

通常通信双方必须用他们的私钥来产生共享密钥，从而可以证明其对私钥的拥有。这种密钥协商提供了更强的认证，保证恶意的一方不能伪装成第三方，通信量少，而且不需要加密和时间戳。

ECMQV 的过程描述如下：

假设 Alice 拥有椭圆曲线密钥对 (A, a)，通过 Bob 拥有 (B, b)，P 为基点，已经通过 CA 认证中心验证其拥有的公钥通过可信方式交换，私钥的所属权也得以证明。Alice 产

生临时会话密钥对(X,x)，x随机产生，$X=xP$。同样地，Bob产生临时会话密钥对(Y,y)，y随机产生，$Y=yP$。通信过程如图12.7所示。

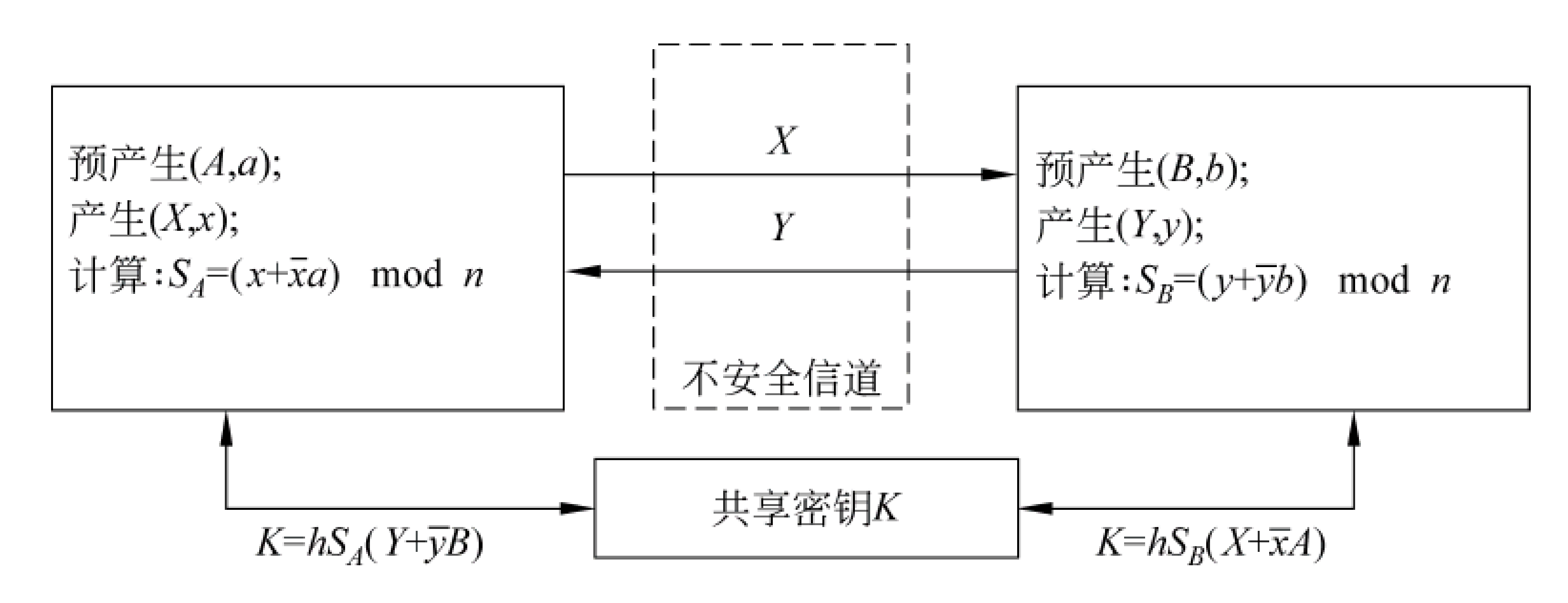


图12.7　ECMQV密钥协商机制

Alice计算$S_A=(x+\bar{x}a)\ \mathrm{mod}\ n$，称为固有签名，$n$为域大小。

Bob计算$S_B=(y+\bar{y}b)\ \mathrm{mod}\ n$。

双方共同计算共享密钥：

$$K=hS_A(Y+\bar{y}B)=hS_B(X+\bar{x}A)$$

这里，h为公因子，$\bar{x}$，$\bar{y}$代表点X或Y的第一个域元素的前L比特，这里$L=\dfrac{\log_2 n+1}{2}$。

安全性：可以保证第三方不能够伪造双方的私钥，因为第三方没有双方的私钥a，b，所以不能够计算出K值。如果双方通信后得到的K值相等，则说明双方的身份可以确定是真实的，密钥对正确无误；否则，双方拒绝密钥对的可靠性。

总之，一般地，设计认证的密钥建立(协商)协议通常会综合运用各种密码学知识，例如：

(1) 数据保密：密钥及数据的传输与存储应该是保密的。

(2) 篡改检测：能检测到对通信的主动攻击。

(3) 身份的识别：在双向通信中，参与通信的双方都能通过身份识别确认通信对方的真实性。

(4) 密钥的新鲜度：保证所使用的密钥不是过时的。

(5) 密钥控制：选择密钥或选择计算机密钥参数的能力。

(6) 密钥的隐式鉴别(implicit key authentication)：保证只有适当的用户才能拥有相应的密钥。

(7) 密钥的确信(key confirmation)：保证适当的用户一定拥有相应的密钥。

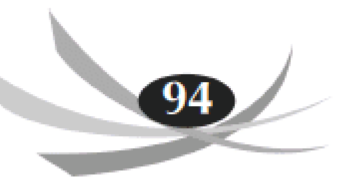
(8) 前向保密性(perfect forward secrecy)：如果有一天，长期使用的密钥泄露了，保证不会泄露以前的会话密钥。

(9) 效率：涉及传输的效率和加密、解密计算的复杂度。

12.2　多方密钥协商*

1221　会议密钥协商

前面主要介绍的是两方密钥协商，有些时候可能需要多方共享的密钥，这种密钥称为



会议密钥(conference key)或者组密钥(group key)。在互联网上的各种协作式应用，如多媒体电子会议、游戏和数据库等中，一个多方共享的密钥会将数据加密、数据完整性和实体认证等服务变得简单有效。如何有效地建立和管理这种多方共享密钥是安全服务的关键。

建立会议密钥的困难性主要包括如下方面：群组成员的加入和离开；群组的合并和分离；叠加问题(在一个成员加入时又有其他成员加入)；通信和计算时间代价；周期地重复建立共享密钥。

下面介绍一种典型的会议密钥协议。该协议中有 n 个用户，允许任意 t 个用户组成的群组通过不安全信道推导出一个共享的会议密钥。这个会议密钥是动态的，而且不同的 t 个用户小组生成的会议密钥不同。

假定 n 个用户中的任意 t 个成员标识符为 $U_0, U_1, \cdots, U_{t-1}$，$p, q$ 是共享的两个大素数，其中 $q|(p-1)$，而 α 是 $\mathbb{Z}_p^*$ 的 q 阶元。

会议密钥的生成协议如下：

(1) 每个用户 $U_i(i=0,1,2,\cdots,t-1)$ 选取一个随机数 $r_i \in \{1,2,\cdots,q-1\}$，计算 $z_i = \alpha^{r_i} \bmod p$，并将 z_i 广播分发给该组的其他 $t-1$ 个成员。

(2) 每个用户 U_i 验证：$z_i^q \equiv 1 \bmod p$。

(3) 每个用户 U_i 接收到用户 U_{i-1}, U_{i+1} 的消息 z_{i-1}, z_{i+1}，计算 $x_i = (z_{i+1}/z_{i-1})^{r_i} \bmod p$，并将 x_i 分发给该组其他 $t-1$ 个成员。

(4) 每个用户 U_i 接收到 $x_j(0 \leqslant j \leqslant t-1$ 且 $j \neq i)$，计算

$$K = K_i = (z_{i-1}^{t \cdot r_i} x_i^{t-1} x_{i+1}^{t-2} \cdots x_{i+(t-2)}) \bmod p$$

正确性验证：

步骤(3)中，

$$\begin{aligned} x_i &= (z_{i+1}/z_{i-1})^{r_i} \bmod p = \alpha^{(r_{i+1}-r_{i-1})r_i} \bmod p \\ &= \alpha^{r_{i+1}r_i - r_i r_{i-1}} \bmod p \end{aligned}$$

步骤(4)中，

$$\begin{aligned} K = K_i &= (z_{i-1}^{t \cdot r_i} x_i^{t-1} x_{i+1}^{t-2} \cdots x_{i+(t-2)}) \bmod p \\ &= \alpha^{tr_i(r_{i-1})} \alpha^{(r_{i+1}r_i - r_i r_{i-1})(t-1)} \alpha^{(r_{i+2}r_{i+1} - r_{i+1}r_i)(t-2)} \cdots \alpha^{r_{i+t-1}r_{i+t-2} - r_{i+t-2}r_{i+t-3}} \end{aligned}$$

为简化，仅观察指数。发现 U_i 计算的 K 为 $K = \alpha^{C_i}$，这里

$$C_i = r_i r_{i-1} + r_{i+1} r_i + r_{i+2} r_{i+1} \cdots r_{i+t-2} r_{i+t-3} + r_{i+t-1} r_{i+t-2}$$

下标 i 的范围为 $[0, t-1]$，故对 t 取模，可发现所有的 U_i 共享的会议密钥即为

$$K = \alpha^{r_0 r_1 + r_1 r_2 + \cdots + r_{t-2} r_{t-1} + r_{t-1} r_0} \bmod p$$

在协议的步骤(2)和步骤(3)中，所有的下标 $i-1, i-2$ 和 $i+1$ 等的计算都是关于模 t 的运算。当协议完成后，所有群组内合法用户都可以计算出相同的会议密钥 $K = \alpha^{r_0 r_1 + r_1 r_2 + \cdots + r_{t-2} r_{t-1} + r_{t-1} r_0} \bmod p$，而群组外的其他人不能获取任何有用的信息。特别要指出的是，当 $t=2$ 时，协议生成的会议密钥为 $K = (\alpha^{r_0 r_1})^2 \bmod p$，恰好是标准 Diffie-Hellman 密钥协商协议建立的共享密钥的平方。该协议的局限是无法抵抗中间人攻击，因为没有包含对实体的认证。

12.2.2　Shamir 三次传递协议

作为一个有趣的扩展，这里介绍一个 Shamir 发明的一个协议，可以使通信双方无须事先交换任何秘密密钥(知道对方的公开密钥)就可进行保密通信。协议中需要用到一个可交换的对称密码：$E_A(E_B(P))=E_B(E_A(P))$。

Alice 的秘密密钥为 A，Bob 的秘密密钥为 B，Alice 想给 Bob 发送一个消息 M，协议如下：

(1) Alice 用自己的秘密密钥加密 M，同时把密文 $C_1=E_A(M)$发送给 Bob。

(2) Bob 用自己的密钥加密 C_1，同时把密文 $C_2=E_B(E_A(M))$发送给 Alice。

(3) Alice 用自己的密钥解密 C_2，同时把结果 $C_3=D_A(E_B(E_A(M)))=E_B(M)$发给 Bob。

(4) Bob 用自己的密钥解密 C_3 恢复明文消息 M。

该协议不能使用“一次一密”作为加密算法(原因留作思考)。Shamir 描述了一个适于该协议的加密算法，类似于 RSA。设 p 是一个大素数，$p-1$ 有一个大的素因子，选择加密密钥 e，使 e 与 $p-1$ 互素。计算 d 使 $de\equiv 1 \bmod (p-1)$。加密函数为 $C=M^e \bmod p$，解密时计算 $M=C^d \bmod p$。该协议不能防止中间人攻击。

第 13 章 高级协议

13.1 比特承诺

1982 年 Blum 提出了比特承诺(bit commitment)的概念[①],1989 年 M. Naor 对比特承诺进行了系统研究,它是密码协议的重要组成部分之一,有很强的应用背景,例如在网上电子投标(拍卖)、电子现金(E-Cash)、电子投票(E-Voting)、在线游戏中。它还是零知识证明、身份识别协议、安全多方计算、盲签名的基础协议。一般地说,任何一个密码协议都可以分解成一系列比特承诺方案。比特承诺方案大都是基于数论中的困难问题。它要解决的问题是:A 向 B 承诺一个预测(可以是一个比特),直到一段时间后才揭示 A 的预测,在这期间,A 不能改变自己的预测。

13.1.1 比特承诺协议概述

一个形象的比喻是:Alice 把要承诺的比特 b 放入一个箱子,用一把只有 Alice 拥有的钥匙才能开启的锁来锁住箱子,然后把这个箱子交给 Bob,当需要向 Bob 证实承诺时,把比特 b 和打开箱子的钥匙交给 Bob,Bob 通过打开箱子可以验证比特 b 的内容没有改动,因为 Bob 相信箱子在 Bob 的保管期间承诺没有被篡改。

比特承诺是构造密码协议的基本组成部分,比特承诺一般包括两个阶段:承诺(commit)阶段和打开(reveal)阶段。在承诺阶段,发送方(承诺者)Alice 选择一个要承诺的比特 b(b 等于 0 或 1),并把能屏蔽该比特的消息 F 发送给 Bob;在打开阶段,Alice 把密钥 K 和 F 发送给接收方(验证者)Bob,Bob 打开 F 并验证 b 是否是 Alice 承诺的比特。

一个安全的比特承诺方案必须满足两个性质:屏蔽性(concealing)和绑定性(binding)。屏蔽性要求在协议的承诺阶段结束时,接收方得不到发送方承诺的比特 b 的值,即使一个不诚实的接收方也要满足这个条件。绑定性要求在打开阶段结束时,接收者只能接受一个合法的承诺,即使发送者试图欺骗时该条件也成立。

一个比特承诺中,可将 Alice 称为一个证明者或承诺者 P,Bob 称为验证者 V。方案中使用到的函数叫做比特承诺函数。比特承诺方案的一个典型框架如下:

步骤 1 比特承诺函数的选定。

设承诺者 P 有一个比特 $b\in\{0,1\}$,X 与 Y 是两个有限集。函数:$f:\{0,1\}\times X\rightarrow Y$ 称为比特承诺函数,如果对从 X 中随机选取的一个元素 x,满足下列性质:

---

① M. Blum. Coin Flipping by Telephone. Proc. of IEEE Sprint COMPCOM. New York: IEEE Press, 1982: 133-137.

(1) 屏蔽性(concealing)。对任意 $b\in\{0,1\}$,验证者不能从 $f(b,x)$确定 b。

(2) 绑定性(binding)。事后承诺者可通过透露 x 的值得到值 $f(b,x)$,使验证者相信 $f(b,x)$是 b 的承诺。前提是承诺者不能找到 x_1,x_2,使得 $f(0,x_1)=f(1,x_2)$。

直观地说,屏蔽性使验证者在 b 被公开前不能从 $f(b,x)$获得 b 的值,绑定性使得承诺者在作出承诺后不能改变承诺的信息。因此,屏蔽性代表承诺者的安全性,绑定性代表验证者的安全性。屏蔽性和绑定性分别代表了双方的不同利益,所以 f 应由双方共同选定,或由可信第三方选定。

步骤 2　比特承诺的承诺阶段。

(1) 承诺者 P 随机选取比特串 x。

(2) P 选定要承诺的比特构成的信息 b。

(3) P 计算出 $f(b,x)$的值 y 并发送给 V。

步骤 3　比特承诺的打开阶段。

(1) P 将(b,x)发送给 V。

(2) V 计算 $f(b,x)$并与 y 比较是否相等。如相等,则 V 接受 P 的承诺,否则拒绝。

可见,比特承诺方案是具有两个阶段的密码协议。

13.1.2　比特承诺方案

1. 利用单向散列函数构造比特承诺方案

承诺者 P 向验证者 V 承诺一个比特 b,可利用单向函数构造如下方案。

承诺阶段:

(1) P 和 V 共同选定某个单向函数 h。

(2) P 随机产生两个比特串:R_1,R_2,P 选定要承诺的比特 b(可能是一个比特或比特串),P 计算单向函数值 $h(R_1,R_2,b)$,并将结果及其中一个随机串,如 R_1,一起发送给 V。

打开阶段:

(1) P 将(R_1,R_2,b)或(R_1,R_2,b)和单向函数 h 一起发送给 V。

(2) V 计算(R_1,R_2,b)的单向函数值,并将该值和承诺阶段第(2)步收到的单向函数值比较(同时比较收到的(R_1,R_2,b)和承诺阶段第(2)步收到的 R_1),检验比特的有效性。

上述协议中,$h(R_1,R_2,b)$和 R_1 是 P 向 V 的承诺证据。P 利用单向函数和随机数阻止 V 对函数求逆以确定承诺的比特。同时,由于单向函数 h 的抗碰撞性,P 找不到 R_2',使得 $h(R_1,R_2,b)=h(R_1,R_2',b')$,从而不可能欺骗 V。

注意:如果 P 不保持 R_2 的秘密性,那么 V 能够计算 $h(R_1,R_2,1)$以及 $h(R_1,R_2,0)$,并比较从 P 接收到的值 $h(R_1,R_2,b)$,从而算出 b。

2. 利用对称密码算法的比特承诺方案

承诺阶段:

(1) P 和 V 共同选定某种对称加密算法 E。

(2) V 产生一个随机比特串 R,并把它发送给 P。

(3) P 首先生成一个想承诺的比特 b(也可能是一个比特串,即承诺一个消息),然后利用对称加密算法对(R,b)进行加密运算得出 $c=E_k(R,b)$,发送 c 给 V。

打开阶段：

(1) P 将密钥 k 及 b 发送给 V。

(2) V 利用密钥 k 解密 c，并利用随机串 R 检验比特 b 的有效性。

上述协议中，c 是 P 承诺的证据，因为 V 没有密钥，无法得知 P 承诺的比特值。如果承诺 $c=E_k(R,b)$ 中不包含 V 给出的随机串 R，则 P 可在承诺后通过使用不同的 k 来解密 c 容易找到不同的 b，从而可以欺骗 V；但是由于承诺 $c=E_k(R,b)$ 中包括了 R，P 要想找到一个新的 b，和随机串 R 一起加密后为 c 的概率是很小的。

比较基于单向散列函数的方案和基于对称密码加密的方案，前者的优势在于不需要验证者发送任何消息。下面给出几个基于单向函数的方案的实例。

例 13.1 基于 Goldwasser-Micali 概率加密体制的比特承诺方案。

承诺函数的单向性基于在不知道 n 的素数因子时，求解模 n 的平方根的困难性。

选定比特承诺函数：

设 $n=pq$ 是两个大素数 p 与 q 之积，t 是模 n 的一个随机选取的平方非剩余。取 $X=Y=\mathbb{Z}_n^*$，$f:\{0,1\}\times X\rightarrow Y$ 为 $(b,x)\rightarrow t^b x^2 \bmod n$。

承诺阶段：

(1) 承诺者 P 随机选取比特串 $x\in\mathbb{Z}_n^*$。

(2) P 选定要承诺的比特 b，计算 $c=f(b,x)=t^b x^2 \bmod n$，发送给验证者 V。

打开阶段：

(1) P 将 t 与 (b,x) 发送给 V。

(2) V 计算 $t^b x^2 \bmod n$，并与 c 比较是否相等以检验承诺的比特 b 的有效性。

屏蔽性分析：因为 x 是随机选择的，所以 $c_0=x^2 \bmod n$ 和 $c_1=tx^2 \bmod n$ 都是 $\mathbb{Z}_p^*$ 中的随机数，V 在不知道 n 的因子分解的情况下不能区分 c_0，c_1 哪一个是平方剩余。

绑定性分析：假设 P 开始承诺 $b=0$，在打开阶段改为 $b=1$，则 P 需要寻找 x'，使得 $x^2=tx'^2$，即 $t=(x/x')^2$，与 t 是平方非剩余的假设矛盾，所以 P 不能改变其作出的承诺。

例 13.2 Petersen 比特承诺协议。

承诺函数的单向性基于离散对数问题的困难性。

选定比特承诺函数：

参与协议的双方在可信第三方的帮助下选择大素数 p 和 $\mathbb{Z}_p^*$ 的生成元 g，从群 $\mathbb{Z}_p^*$ 中随机选择元素 $t\in\mathbb{Z}_p^*$。$X=Y=\mathbb{Z}_p^*$，$f:\{0,1\}\times X\rightarrow Y$ 为 $(b,x)\rightarrow t^b g^x \bmod p$。

承诺阶段：

(1) 承诺者 P 选择所需的承诺比特 b，并产生随机数 $x\in\mathbb{Z}_p^*$。

(2) P 计算 $c=f(b,x)=t^b g^x \bmod p$，发送 c 给 V。

打开阶段：

(1) P 将 b 和 x 发送给 V。

(2) V 验证 c 是否与收到的承诺一致，如果一致，认为承诺有效，否则无效。

屏蔽性分析：这是因为 x 是随机选择的，所以 $c_0=g^x \bmod p$ 和 $c_1=tg^x \bmod p$ 都是 $\mathbb{Z}_p^*$ 中的随机数，V 不能区分 c_0，c_1。

绑定性分析：假设 P 开始承诺 $b=0$，在打开阶段想改为 $b=1$，则 P 需要寻找 x'，使得



$g^x = tg^{x'}$，即 $t = g^{x-x'}$。这意味着P需要计算随机数 t 的离散对数，而这对P来说是计算困难的，所以P不能改变其作出的承诺。

思考 13.1 如何将比特承诺扩展为消息承诺，即一次承诺多个比特？

使用 m 或者 $h(m)$（当 m 过大时）代替承诺函数中的 b 即可。对其分析留作练习。

更一般的方法是使用伪随机发生器。Noar 提出使用伪随机发生器构造比特承诺方案。协议如下。

选定比特承诺函数：

参与协议的双方在可信第三方的帮助下选择伪随机发生器 G，再选择随机串 R，R 的长度足够大，如 128bit。

承诺阶段：

（1）P选择所需要的承诺比特 b，产生随机数 s，作为伪随机发生器所需的种子。

（2）P计算：如果 $b=0$，则 $c=G(s)$；如果 $b=1$，则 $c=G(s)\oplus R$。P将承诺 c 发送给V。

打开阶段：

（1）P将 s 和 b 发送给V。

（2）V验证 c 的计算是否与收到的承诺一致，如果一致，则承诺有效，否则无效。

屏蔽性分析：在获取 b 之前，由于 G 是伪随机发生器，V 无法区分 $G(s)$ 和 $G(s)\oplus R$。

绑定性分析：假设P开始承诺 $b=0$，打开时想改为 $b=1$，需要寻找 s'，使得 $G(s')\oplus R=c=G(s)$，由伪随机发生器的性质知，这是困难的。

思考 13.2 使用伪随机发生器构造比特承诺协议时想对消息 m 承诺，如何实现？

改变承诺函数为 $c=G(s)\oplus(R\cdot m)$。对其分析留作练习。

13.1.3 基于离散对数问题的承诺方案

13.1.1 节和 13.1.2 节主要给出比特承诺方案，本节讨论对多个比特的承诺。下面介绍一个著名的基于离散对数问题是困难的这一假设的承诺方案。

选定承诺函数：

设 p 是一个大素数，g_0，h_0 是 $\mathbb{Z}_p^*$ 的两个不同的生成元。承诺者希望承诺的秘密是 v。

承诺阶段：

（1）承诺者均匀地选择一个随机数 $r\in\mathbb{Z}_p^*$，并计算承诺

$$c \equiv g_0^v h_0^r \bmod p$$

（2）承诺者将 c 发送给接收者作为对数据 v 的承诺。

打开阶段：

承诺者将 v，r 发送给接收者。接收者验证下述等式是否成立：

$$g_0^v h_0^r \equiv c \bmod p$$

若成立，则接受承诺值 v，否则拒绝。

屏蔽性分析：需要证明对 v 和 v' 的两种承诺是不可区分的，即 $g_0^v h_0^r \bmod p$ 和 $g_0^{v'} h_0^r \bmod p$ 不可区分。假设 $v'=v+b$，则需要证明 $g_0^v h_0^r \bmod p$ 和 $g_0^{v+b} h_0^r \bmod p$ 不可区分。根

据生成元的性质，有 $g_0^b \in \mathbb{Z}_p^*$，又由于 h_0 是生成元，于是一定存在一个 s 使得 $h_0^s \bmod p = g_0^b \bmod p$，于是需要证明的实际上是 $g_0^v h_0^r \bmod p$ 和 $g_0^v h_0^{r+s} \bmod p$，由于 r 是随机数，故两者是不可区分的。

绑定性分析：需要证明不能找到 v'，使得 $g_0^{v'} h_0^{r'} \bmod p = c \bmod p = g_0^v h_0^r \bmod p$。假设敌手可以找到 v'，r' 使等式满足，于是敌手可以采用如下方式计算离散对数 $\log_{h_0} g_0 \bmod p$：

$$g_0^{v'-v} \equiv h_0^{r-r'} \bmod p$$
$$g_0 \equiv h_0^{(r-r')/(v'-v)}$$
$$\log_{h_0} g_0 = (r-r')/(v'-v)$$

这显然违背了离散对数是困难的这一假设。

13.1.4 电话投币协议

电话投币(telephone coin flipping)协议是比特承诺协议的一种应用。

首先看一个问题：假设 Alice 住在北京，Bob 住在上海，他们想通过电话投掷硬币来决定谁能够获得一辆车。Alice 打电话给 Bob 说通过电话来投硬币决定。Bob 选择了反面，Alice 说：“对不起，是正面。”于是 Alice 得到了那辆车。Bob 怀疑 Alice 有不诚实的行为，但是不知道如何解决。

对于上面这个电话投币问题，设想用一个抛币落井的方法来解决。设想有一口清澈的深水井，Alice 站在水井的旁边，Bob 远离这口井，Alice 将硬币抛进水井里，硬币落在水井里，现在 Alice 能看到水井里的结果，但 Alice 不能到水井里去改变硬币的状态(如正反面)。当 Alice 将硬币抛进水井里，Bob 不能看见水井里的硬币，只有当 Bob 猜完硬币的状态后，Alice 才让 Bob 走近，看到井底的硬币。这样，Alice 不能欺骗 Bob。原因是 Alice 在 Bob 猜测之前有一个承诺，不能改变。于是，很自然地会想到利用安全散列函数。

1. 利用安全散列函数抛硬币

设 Alice 和 Bob 都知道某一个安全散列函数 H。(Alice 投，Bob 猜测。)

(1) Alice 选择一个随机数 r，计算 $y=H(r)$。

(2) Alice 将 y 发送给 Bob。

(3) Bob 猜测 r 是偶数还是奇数，并将猜测结果发送给 Alice。

(4) Alice 公布此次抛币的结果，并将 r 发送给 Bob。

(5) Bob 检查 $y=H(r)$。

该协议的安全性取决于 H 的安全性。Bob 通过 y 判断 r 的奇偶性必须是困难的，否则 Bob 总能猜对。同时，Alice 在抛币后不能改变，即 Alice 无法找到 r 和 r'，满足 r 和 r' 的奇偶性不同，但有 $H(r)=H(r')$，否则 Alice 总能欺骗 Bob。

于是，这个抛币协议具有如下性质：Alice 必须在 Bob 猜测之前投币(承诺)，在 Bob 猜测之后 Alice 不能再投币(绑定)。Bob 无法知道硬币的正反面，只能依靠随机猜测。

2. 采用平方根投硬币(Bob 投，Alice 猜测)

(1) Alice 选择两个大素数 p，q，将乘积 $n=pq$ 发送给 Bob。

(2) Bob 在 1 和 $n/2$ 之间随机选择一个整数 u，计算 $z=u^2 \bmod n$，并将 z 发送给 Alice。

(3) A 计算模 n 下 z 的 4 个平方根 $\pm x,\pm y$（A 知道 n 的分解，所以可以计算）。设 x' 是 $x \bmod n$，$-x \bmod n$ 中的较小者，y' 是 $y \bmod n$，$-y \bmod n$ 中的较小者，则由于 $1<u<n/2$，所以 u 为 x'，y' 之一。

(4) A 猜测 $u=x'$ 或 $u=y'$，或者 A 找出最小的 i 使得 x' 的 i 个比特与 y' 的第 i 个比特不同，A 猜测 u 的第 i 个比特是 0 还是 1。A 将猜测发送给 B。

(5) B 告诉 A 猜测正确或不正确，并将 u 发送给 A。

(6) A 公开 n 的因子。

因为 u 是 B 随机选取的，A 不知道 u，所以要猜测 u 只能是计算模 n 下 z 的 4 个平方根，猜中的概率为 1/2。再考虑 B 是否能欺骗 A，如果 B 在 A 猜测完之后能够改变 u 的值，则 A 的猜测总是不正确，但是 B 必须保持 $z=u^2 \bmod n$，于是 B 只能是在 x'，y' 之间相互改变，这意味着 B 掌握了 x'，y'，于是可以通过 $\gcd(x'-y',n)$ 或者 $\gcd(x'+y',n)$ 求出 p 和 q，这与 n 这一大整数分解问题是困难的相矛盾。

3. 利用二次剩余投币

设 n 是两个大素数 p 和 q 的乘积，即 $n=pq$。整数 a 满足 $0<a<n$ 和 $\gcd(a,n)=1$，则其中一半的 a 有 Jacobi 符号 $(a/n)=1$。在满足 $(a/n)=1$ 的所有 a 中，只有一半是模 n 的平方剩余，而判断 a 是否为模 n 的平方剩余与分解 n 是等价的。

下面给出协议（B 投 A 猜）。

(1) B 选择 p,q，计算 $n=pq$，再选择满足 $(a/n)=1$ 的随机数 a，将 n 和 a 发送给 A。

(2) A 猜测 a 是模 n 的平方剩余或平方非剩余，并将结果告诉 B。

(3) B 告诉 A 猜测正确或不正确，并将 p,q 发送给 A。

(4) A 检查 p,q 都是素数，且 $n=pq$。

显然 A 猜中的概率是 1/2。协议执行完后，A 根据 p,q 可求出 $a \bmod n$ 的 4 个平方根（如果 a 是模 n 的平方剩余），以检查 B 是否在 A 猜测完后对结果进行了修改。

13.2 零知识证明协议

零知识证明（Zero Knowledge Proof，ZKP）是构造安全的密码学协议的主要工具，ZKP 是由 S. Goldwasser、S. Micali 以及 C. Rackoff 等在 1985 年首先提出的，在密码协议的设计和分析中占有重要的地位。

零知识证明是一种密码协议，和比特承诺协议类似，该协议的一方称为证明者（prover），通常用 P 表示，协议的另一方是验证者（verifier），一般用 V 表示。零知识证明是指 P 试图使 V 相信某个论断是正确的，但不向 V 提供任何有用的信息，或者说在 P 论证的过程中，V 得不到任何有用的信息。也就是说，零知识证明除了证明了 P 的论断的正确性外，不泄露任何其他信息或者知识。因此，可以放心地使用该协议。例如，口令身份验证协议中存在的一个问题就是，证明者向验证者证明自己知道口令的同时把口令泄露给了验证者。这样验证者就可以在其他场合冒充证明者，使证明者处于不利地位。

零知识证明起源于最小泄露证明。在交互证明系统中，P 知道某个秘密，向 V 证明自己掌握这个秘密，但又不向 V 泄露这一秘密，这就是最小泄露证明。如果 V 除了知道

P 掌握秘密外，不能得到任何其他信息，则是零知识证明。

思考 13.3 零知识证明是否和完善保密具有某种类比关系？

完善保密是针对加密方案的安全性，从密文无法推知明文的任何信息。零知识证明是针对协议方案的安全性，验证者无法推知除需要证明的信息外的任何信息。可见，两者之间具有某种“类似性”。

零知识证明可根据交互性分为交互零知识证明和非交互零知识证明。零知识交互证明系统(Zero Knowledge Interactive Proof, ZKIP)的模型如下：假设 P 和 V 是两台图灵机，P 采用交互式证明向 V 证明一个断言 S，而最终 V 除了相信 S 外，得不到任何额外信息。

按照该模型，ZKIP 必须满足以下 3 个特性：

(1) 完备性(completeness)：如果 P 证明 S 为真，则 V 拒绝接受 S 的概率非常小。

(2) 合理性(soundness)：如果 P 有欺骗行为，则 V 接受 S 的概率非常小。

(3) 零知识性(zero-knowledge)：V 除了相信 S 外，不能获得额外信息。

为了更好地理解这 3 个特性，先看下面几个例子。

13.2.1 零知识证明的 3 个经典示例

J. J. Quisquater 与 T. Berson 等人在 1989 年的 Crypto'89 会议上给出了一个典型的例子，即用洞穴的故事来说明零知识证明协议。

O. Goldreich 等在 1986 年给出了一个证明图同构的例子。

M. Blum 在 1986 年给出了一个证明 Hamilton 回路问题的例子。

例 13.3 阿里巴巴的洞穴故事。

如图 13.1 所示，洞穴中有一个秘密通道门位于 C、D 之间，只有知道秘密咒语的人(阿里巴巴)才可以打开这扇门。假设 P 知道这个咒语，他要向 V 证明自己知道这个咒语，但又不向 V 泄露这个咒语，那么 P 与 V 可以通过下面的游戏来达到目的。

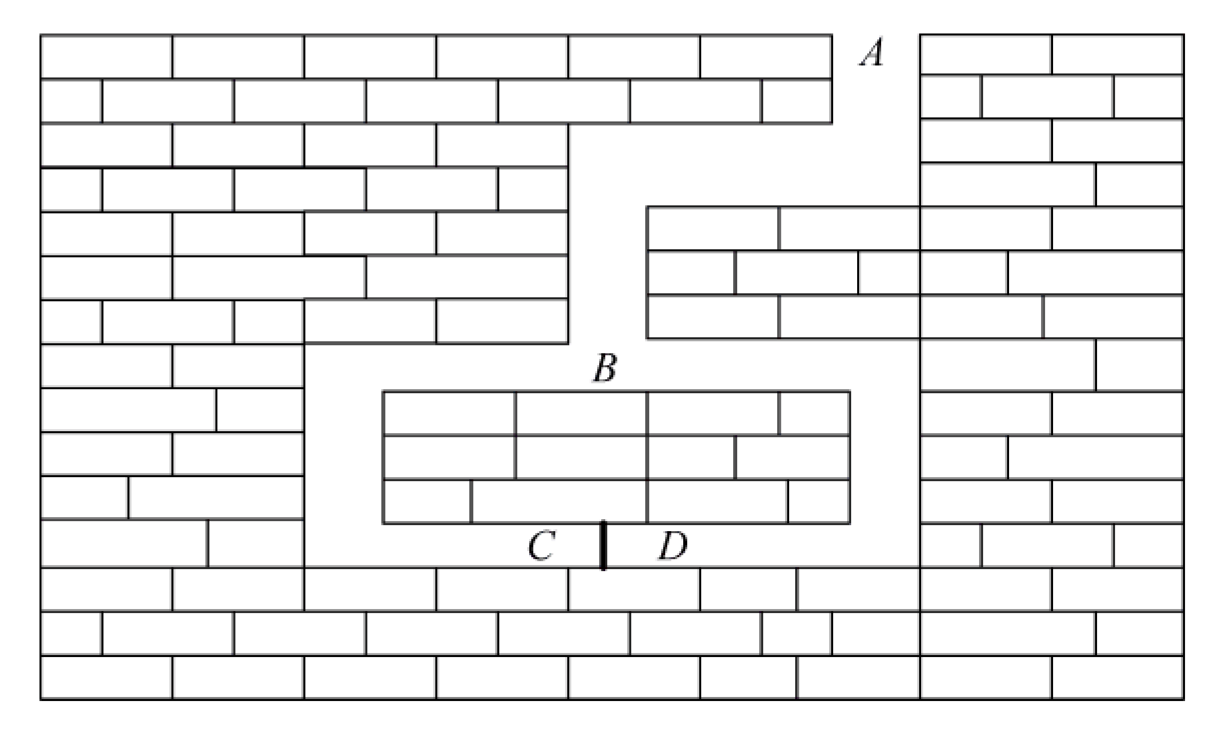


图 13.1 洞穴中秘密咒语的零知识证明

(1) V 站在 A 位置。

(2) P 从位置 A 出发经过 B 走进洞穴，到达 C 或 D 位置。

(3) 当 P 消失在洞穴后，V 走到 B 位置。

(4) V 随机命令 P 从左通道或从右通道返回 B 位置。

(5) P 按照 V 的命令从左通道或右通道返回 B 位置，在必要时 P 使用咒语打开 C 与 D 位置之门。

(6) P 与 V 重复这个游戏若干次。

完备性分析：在上述游戏中，如果 P 知道咒语，则总能成功。

合理性分析：如果 P 不知道咒语，那么 P 每次只能按照原路线从 C 或 D 返回到 B 位置。这样，P 每次成功按 V 的要求从洞穴深处返回到 B 的概率是 1/2，所以 P 能 n 次都成功按 V 的要求返回到 B 位置的概率为 $1/2^n$。当 n 较大时，n 次都成功的概率就非常小。因此，游戏重复多次后，若 P 均能成功，则 V 完全可相信 P 知道打开 C 与 D 之间那扇门的秘密咒语。

零知识性分析：上述游戏证明 P 对秘密咒语的拥有，且没有透露任何关于秘密咒语的信息(即零知识性)。

下面看一个证明图同构的例子。

例 13.4 证明图同构。

设 $G_1=\{\{a_1,a_2,\cdots,a_n\},E_1\}$ 与 $G_2=\{\{a_1,a_2,\cdots,a_n\},E_2\}$ 是具有相同顶点集合 $\{a_1,a_2,\cdots,a_n\}$ 但边集合 E_1 和 E_2 不同的两个图。一般来说，对于输入规模 n 充分大的两个图，要证明它们是否同构是一个 NP 完全问题(G_1,G_2 同构是指从 G_1 的顶点到 G_2 的顶点集合之间存在一个双射 π，当且仅当 x,y 是 G_1 上的相邻点时，$\pi(x)$ 和 $\pi(y)$ 是 G_2 上的相邻点)。

假设 P 知道 G_1 和 G_2 是同构的，当 V 不知道 G_1 和 G_2 是同构的。现在 P 要向 V 证明 G_1 和 G_2 是同构的，但又不想告诉 V 如何证明同构关系(即给出图 G_1 和 G_2 的点集对应关系)，那么 P 与 V 进行如下步骤来达到目的(图 13.2)：

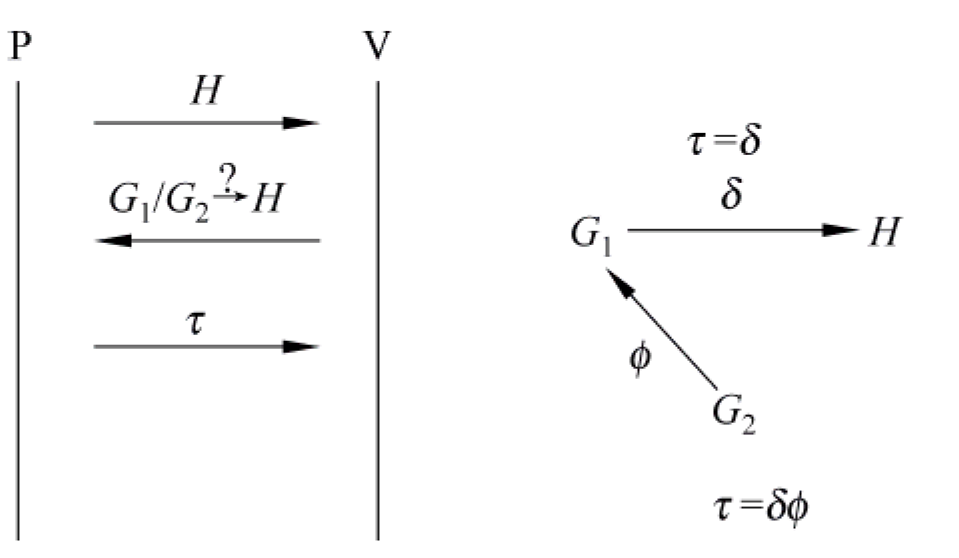


图 13.2 图同构的零知识证明

(1) P 随机选取 $\{a_1,a_2,\cdots,a_n\}$ 上的一个置换 δ，在 δ 作用下，图 G_1 变换成 H，即 $H=G_1^{\delta}$。P 将 H 告诉 V。

(2) V 随机地要求 P 证明：G_1 与 H 同构，或者 G_2 与 H 同构。

(3) P 为完成 V 的要求，先找定 $\{a_1,a_2,\cdots,a_n\}$ 的一个置换 τ，将该置换告诉 V。置换 τ 是这样找的：当 V 要求 P 证明 G_1 与 H 同构时，P 令 $\tau=\delta$；当 V 要求 P 证明 G_2 与 H 同构时，P 令 $\tau=\delta\phi$(其中 ϕ 是 P 事先已知的 G_1 与 G_2 之间的同构映射)。

(4) V 验证在置换 τ 作用下图 G_1 或 G_2 是否能置换成 H。

(5) P 与 V 重复执行(1)～(4)若干次。

完备性分析：如果 P 知道同构映射 ϕ，则总能回答成功。

合理性分析：如果 P 不知道同构映射 ϕ，则只能以 1/2 的概率回答成功(即刚好 V 要求给出 G_1 到 H 的映射时 P 给出 $\tau=\delta$)。若 n 次都能回答成功，则 P 在不知道同构映射 ϕ 下成功欺骗的概率为 $1/2^n$。当 n 很大时，该概率非常小。

零知识性分析：证明过程中 V 没有获得关于 G_1 和 G_2 之间同构映射 ϕ 的信息。其实，这里可以把 $\tau=\delta\phi$ 理解为对 ϕ 用 δ 进行"一次一密"的加密。

例 13.5 证明 Hamilton 路径问题(Hamilton Path Problem，HPP)。

在图论中，能够遍历图 G 的每个顶点的路径称为 Hamilton 路径，如果一个图包含一条 Hamilton 路径，则称为 Hamilton 图。迄今为止尚未找到判断 Hamilton 图的充要条件，构造图 G 的 Hamilton 路径是一个 NP 完全问题。假设证明者 P 掌握的信息是图 G 的 Hamilton 路径，并希望向验证者 V 证明这一事实，协议执行如下：

(1) P 随机地构造一个与图 G 同构的图 G' 并发送给 V。

(2) V 随机地要求 P 做下述两个任务之一：证明图 G 和图 G' 同构，或者指出 G' 的一条 Hamilton 路径。

(3) P 按照 V 的要求做下述两个工作之一：一是证明图 G 和图 G' 同构，但不指出图 G 和图 G' 的 Hamilton 路径；二是可以指出图 G' 的 Hamilton 路径，但不证明图 G 和图 G' 同构。

(4) 重复执行上述(1)～(3)若干次。

容易知道满足完备性。

合理性分析：如果 P 不知道图 G 的 Hamilton 路径，则只有 1/2 的概率完成任务，n 次重复后成功的概率为 $1/2^n$。当 n 很大时，该概率非常小。

零知识性分析：在第(3)步中，即使 P 向 V 指出图 G' 的一条 Hamilton 路径，V 也没有得到任何关于图 G 的 Hamilton 路径的信息，因为求两个图的同构并不比求一个图的 Hamilton 路径简单。

13.2.2 基于困难问题构造零知识证明

零知识证明可以用于构造身份证明协议，即某个声称者证明其身份的过程就是他证明其拥有某个秘密知识的过程。其中协议完备性、合理性、零知识性的论证留作练习。

1. 基于大整数分解的零知识证明

设 p 和 q 是两个大素数，$n=pq$，假设 P 知道 n 的因子。如果 P 想让 V 相信他知道 n 的因子，且 P 不想让 V 知道 n 的因子，则 P 和 V 可以执行下面的协议：

(1) V 随机选择一个大整数 x，计算 $y=x^4 \bmod n$，将结果 y 告诉 P。

(2) P 计算 $z=\sqrt{y} \bmod n$，P 将结果 z 告诉 V。

(3) V 验证 $z=x^2 \bmod n$ 是否成立。

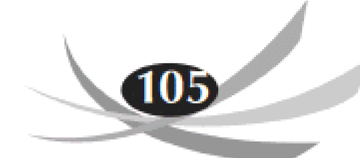
(4) 重复(1)～(3) t 次。

合理性在于计算 $\sqrt{y} \bmod n$ 等价于对 n 进行因子分解。

2. 基于离散对数的零知识证明

P 想向 V 证明他知道满足方程 $\alpha^x=\beta \bmod p$ 的 x。其中 p 是一个大素数，x 是与 p 互素的随机数。α,β,p 是公开的，x 是保密的。P 在不泄露 x 的信息的情况下，向 V 证明他知道 x 的过程如下：

(1) P 选择随机数 $r(0<r<p-1)$，计算 $h=\alpha^r \bmod p$，将 h 发送给 V。



(2) V 随机选择一个整数 $b=0$ 或 $b=1$ 发送给 P。

(3) P 计算 $s=(r+bx) \bmod (p-1)$，并发送给 V。

(4) V 验证 $\alpha^s=h\beta^b \bmod p$。

(5) 重复(1)～(4) t 次。

其实，细心的读者会发现，这一过程和 ElGamal 签名有神似之处。

13.3　不经意传输

先思考两个问题：

(1) 设 A 有一个秘密，想以 1/2 的概率传递给 B，即 B 有 50%的机会收到这个秘密，另外 50%的机会什么也没有收到，协议执行完后，B 知道自己是否收到了这个秘密，但 A 却不知道 B 是否收到了这个秘密。

(2) A 是机密的出售者，A 列举了很多问题，想出售这些问题的答案，B 想购买其中的一个或几个问题的答案，但不想让 A 知道自己买的是哪些问题的答案。

这两个问题在网上交易中会体现为类似的安全需求，如用户 B 从用户 A 接收消息，但出于隐私保护的考虑，用户 B 不想让用户 A 知道他到底接收的是哪条消息。例如网上订购，消费者可能不愿意暴露所购买的商品；在线付费浏览，用户可能不希望暴露浏览的敏感信息。

13.3.1　不经意传输协议概述

上述问题可以利用不经意传输(Oblivious Transfer，OT)解决。Rabin 于 1981 年提出 OT 的概念。OT 是一种可保护隐私的两方通信协议，使通信双方以选择模糊化的方式传输信息。该协议使得服务接收方以不经意的方案得到服务发送方输入的某些信息，这样可以保护接收方的隐私不被发送方知道。

不经意传输协议有着丰富的实际应用，如隐私保护的信息检索、不经意抽样、公平的电子合同的签订、内容保护等。它也可作为基本组件构造其他安全协议，如比特承诺、零知识证明、安全多方计算以及电子支付协议等。它是零知识证明协议在实际中的一个十分重要的应用形式。它也是一种特殊类型的比特承诺协议，发送方对一系列值给出承诺，接收方在收到发送方给出的承诺后选中其中之一作为确认，当发送方并不知道接收方选中的承诺对应的是哪一个值。

不经意传输协议也可译为“健忘”传输协议，它是从一个消息集合秘密获取部分信息的一种重要方法。所谓“健忘”传输是指发送方以 50%的概率传送一个秘密给接收方，接收方有 50%的机会收到这个秘密，有 50%的机会什么也没有收到。协议执行完毕后，接收方知道他是否收到这个秘密，但发送方却不知道。

一般地说，不经意传输协议是一个由发送方 A 和接收方 B 参与的两方通信协议，其基本思路是：发送方 A 发出 n 条消息 $m_1,m_2,\cdots,m_n$，执行协议后接收方 B 将得到其中的一条或几条消息。发送方 A 不能控制接收方 B 的选择，也不知道接收方 B 收到的是哪些具体消息；而接收方 B 不能得到其选择之外的消息。

不经意传输协议在设计时需要考虑到以下因素：

(1) 协议的参与者。协议的参与者是发送方 A 与接收方 B。如果一个参与方按步骤执行协议，但试图从接收到的消息计算出额外的信息，则称为半可信的；如果一个参与方任意背离协议以获得额外信息，则称其为恶意的。

(2) 正确性。若发送方 A 与接收方 B 正确执行协议，则 B 可得到其所选择的消息。

(3) 不经意性(接收者的隐私性)。接收方 B 的不同选择所对应的传送副本对于发送方 A 是不可区分的，A 无法得知 B 究竟选择了哪些消息。

(4) 安全性(发送者的隐私性)。接收方 B 不能得到他没有选择的消息。

在 Rabin 提出 OT 概念之后，不经意传输又出现了多种形式：

(1) 二选一(1-out-of-2)OT。A 有两个消息 m_1, m_2，协议执行完后 B 得到其中一个消息(A 的隐私性)，A 不知道 B 选择的是哪一个消息(B 的隐私性)，B 可以确信自己得到了想要的消息(正确性)。

(2) 多选一(1-out-of-n)OT。A 有 n 个消息 $m_1, m_2, \cdots, m_n$，协议执行完后 B 得到其中的一个消息(A 的隐私性)，A 不知道 B 选择的是哪一个消息(B 的隐私性)，B 可以确信自己得到了想要的消息(正确性)。

(3) n 选 k(k-out-of-n)OT。A 有 n 个消息 $m_1, m_2, \cdots, m_n$，协议执行完后 B 得到其中的 k($k<n$)个消息(A 的隐私性)，A 不知道 B 选择的是哪一个消息(B 的隐私性)，B 可以确信自己得到了想要的消息(正确性)。

另外，从执行过程可将 OT 划分为交互式 OT 和非交互式 OT。从实现方法上，有经典方法(基于计算复杂性的 OT，基于信息论的 OT)和基于量子理论的 OT。

13.3.2 不经意传输协议的设计

1. 基于大整数分解的二选一 OT

假设 A 想通过不经意传输协议传输给 B 的秘密是整数 n(n 为两个大 Blum 素数之积)的因子分解。这个问题具有普遍意义，因为任意秘密都可以通过 RSA 加密算法传送给相应的解密者，得到 n 的因子分解就可成为解密者，从而得到秘密。

协议描述如下：

(1) A 选择形式为 $4k+3$ 的两个大素数(Blum 素数)，发送这对素数 p, q 的乘积 $n=pq$ 给 B，但将 p, q 保留为自己的秘密。

(2) B 随机选取一个整数 x($0<x<n$)，且 $\gcd(x,n)=1$，即 x 是比 n 小且与 n 互素的正整数，然后发送 $a=x^2 \bmod n$ 给 A。

(3) A 根据已知的 p, q 求出 $x^2=a \bmod p$ 和 $x^2=a \bmod q$ 对应的两个根，然后 A 随机选取其中的一个根发送给 B。

当 n 为 Blum 整数时，求 $x^2=a \bmod p$ 是容易的，即 $x=\pm a^{\frac{p+1}{4}} \bmod p$，同理可得 $x^2=a \bmod q$ 的根。求出根后，可根据中国剩余定理求出 $x^2=a \bmod n$ 的 4 个根：$x, n-x, y, n-y$。

如果 B 收到的是 y 或者 $n-y$，则 B 通过已知的 x 和接收的 y 就可以确定出 p, q：$\gcd(x+y,n)=p$ 或 $\gcd(x+y,n)=q$。如果 B 收到的是 x 或 $n-x$，则得不到任何有用信息。

显然,B得到因子分解的概率为1/2。

2. 基于离散对数问题的二选一OT

与前面的协议不同,下面这个协议是非交互的,即B不向A发送任何消息。设用户都知道一个大素数 p,$\mathbb{Z}_p^*$ 的生成元 g 和另一个大素数 c,但无人知道 c 的离散对数。

B按如下方式产生公钥和私钥:随机选取一个比特 i 和一个数 $x(0\leqslant x\leqslant p-2)$,计算 $y_i=g^x$,$y_{1-i}=c(g^x)^{-1}$,(y_0,y_1) 为公钥,以 (i,x) 为私钥。由于B不知道 c 的离散对数,所以只知道 y_0 或 y_1 的离散对数。A不知道 y_0,y_1 中哪个数的离散对数(即 x)是B已知的。A可通过 $y_0y_1=c$ 验证B的公钥是否正确。

设A的两个秘密是 s_0,s_1,是二进制数。$\oplus$ 为异或运算,若操作的两数不等长,必要时可以在较短的数前补0。

协议描述如下:

A在0到 $p-2$ 之间随机取两个整数 k_0,k_1,对 $j=0,1$ 计算

$$c_j=g^{k_j},\quad d_j=y_j^{k_j},\quad m_j=s_j\oplus d_j$$

将 c_0,c_1,m_0,m_1 发送给B。

B用自己的私钥计算 $c_i^x=g^{xk_i}=y_i^{k_i}=d_i$,$s_i=m_i\oplus d_i$。由于B不知道 y_{1-i} 的离散对数,所以无法得到 d_{1-i},s_{1-i}。

这一协议的基本思想是:B拥有两个"公钥"(y_0,y_1),一个是可以用私钥 x 解密的,另一个是无法用私钥解密的。A用两个"公钥"进行加密,然后发送给B,B则只能解密其中之一。由于A不知道B能解密哪一个,所以不经意性满足。接收方B只能得到二者之一,安全性(发送者的隐私性)满足。B不能通过制造两个都有效的公钥来欺骗,因为A可以验证 $y_0y_1=c$。

一般地,可以得到Even二选一OT协议[①],这是Even等在1985年提出的。

协议描述如下:

(1) 发送方A选择公钥系统 (E_x,D_x),并选择两个随机数 $c_0,c_1\in u_x$,u_x 是上述公钥系统的消息空间。A把公钥 E_x 和随机数 c_0,c_1 传送给B。

(2) B选择随机数 $r\in\{0,1\}$ 以及会话密钥 $k\in u_x$。B用A的公钥加密会话密钥 k,同时用A给的两个随机数之一来加密密文,即 $q=E_x(k)\oplus c_r$,最后将 q 发送给A。

(3) A用私钥 D_x 解密 q。对于 $i=0,1$,A分别计算 $k_i'=D_x(q\oplus c_i)$,得到两个结果,一个是真正的会话密钥 k,而另一个是无意义的随机数,但A无法区分。

(4) A分别使用在上一步中产生的两个密钥(一个真的会话密钥,一个假的会话密钥)加密他的两份消息 m_0,m_1,并把两份消息都发送给B。

(5) B收到一份用正确的会话密钥加密的消息及一份用假的会话密钥加密的消息。当B用会话密钥解密每一份消息时,能得到其中之一,另一份则是无意义的。

该协议的设计思想是:避免设计两个"公钥"的烦琐,而是采用两个"会话密钥"(一个有效,一个无意义)来加密传递的消息,达到二选一OT的目的。但这个方案需要交互。

---

① S. Even, O. Goldreich, A. Lempel. A Randomized Protocol for Signing Contracts. Communications of the ACM, 1985, 28(6): 637-647.

3. 基于单向函数的多选一 OT

设 A 有多个秘密，想将其中之一传递给 B，使得只有 B 知道 A 传递的是哪个秘密。设 A 的秘密是 $s_1, s_2, \cdots, s_k$，每一个秘密是一比特序列。

协议描述如下：

(1) A 告诉 B 一个单向函数 f，但对 f^{-1} 保密。

(2) 设 B 想得到秘密 s_i，在 f 的定义域内随机选取 k 个值 $x_1, x_2, \cdots, x_k$，将 k 元组 $(y_1, y_2, \cdots, y_k)$ 发送给 A，其中

$$y_j = \begin{cases} x_j, & j \neq i \\ f(x_i), & j = i \end{cases}$$

(3) A 计算 $z_i = f^{-1}(y_j)(j=1,2,\cdots,k)$，将 $z_j \oplus s_j (j=1,2,\cdots,k)$ 发送给 B。

(4) 由于 $z_i = f^{-1}(y_i) = f^{-1}(f(x_i)) = x_i$，所以 B 知道 z_i，因此可以从 $z_i \oplus s_i$ 获得 s_i。

由于 B 没有 $z_i(j \neq i)$ 的信息，因此无法得到 $s_j(j \neq i)$，而 A 不知道 k 元组 $(y_1, y_2, \cdots, y_k)$ 中哪个是 $f(x_i)$，因此无法确定 B 得到的是哪个秘密。

但是，如果 B 不遵守协议，用 f 对多个 x_j 求得 $f(x_j)$，就可获得多个秘密。

4. 基于离散对数的多选一 OT

2004 年 Tzeng 等提出一个基于 DDH 假设(见 14.1.2 节)的多选一 OT 协议。

G 是一个 q 阶循环群，g, h 是 G 的两个生成元，$\log_g h$ 保密。发送者 A 的输入为 $m_1, m_2, \cdots, m_n \in G$，接收者 B 的选择为 $a, 1 \leqslant a \leqslant n$。

协议描述如下：

(1) B 发送 $y = g^r h^a, r \in \mathbb{Z}_q$。

(2) A 发送 $c_i = (g^{k_i}, m_i (y/h^i)^{k_i}), k_i \in \mathbb{Z}_q, 1 \leqslant i \leqslant n$。

(3) B 由 $c_a = (s, t)$，计算 $m_a = t/s^r$。

其设计思想是：B 对选择的序号 a 进行了承诺，且序号 a 具有隐蔽性。A 将消息用包含承诺 a 的密钥进行加密，只有序号 a 对应的密文才能解密出消息。

最后，讨论二选一 OT、多选一 OT 和 n 选 k OT 间的关系。多选一 OT 可以通过调用二选一 OT 来设计，n 选 k OT 可以通过调用二选一 OT、多选一 OT 来实现。一个直观地实现 n 选 k OT 的方法是将多选一 OT 运行 k 次，但这需要 k 倍于多选一 OT 的通信和计算代价。因此，协议设计的目标不仅是完成功能，而且是设计通信负担小、计算代价低的协议。

13.4 秘密共享

13.4.1 秘密共享概念的提出

首先思考两个场景：①某个银行有 3 位出纳，他们每天都要开启保险库。为防止每位出纳可能出现的监守自盗行为，银行规定至少有两位出纳在场才能开启保险库。②核按钮通常掌握在三方手上：总统、国防部长、国防部，只有当三方中的两方在场时，才可以最终控制核按钮。

上述两个问题可以利用秘密共享方案(secret sharing scheme)来实现。所谓秘密分享就是将一个密钥分成许多份额(share,又叫影子,shadow),然后秘密地分配给一些有关人员,使得某些有关人员在同时拿出他们的份额后可以重建密钥,而另外一些有关人员在同时拿出他们的份额后不能重建密钥。

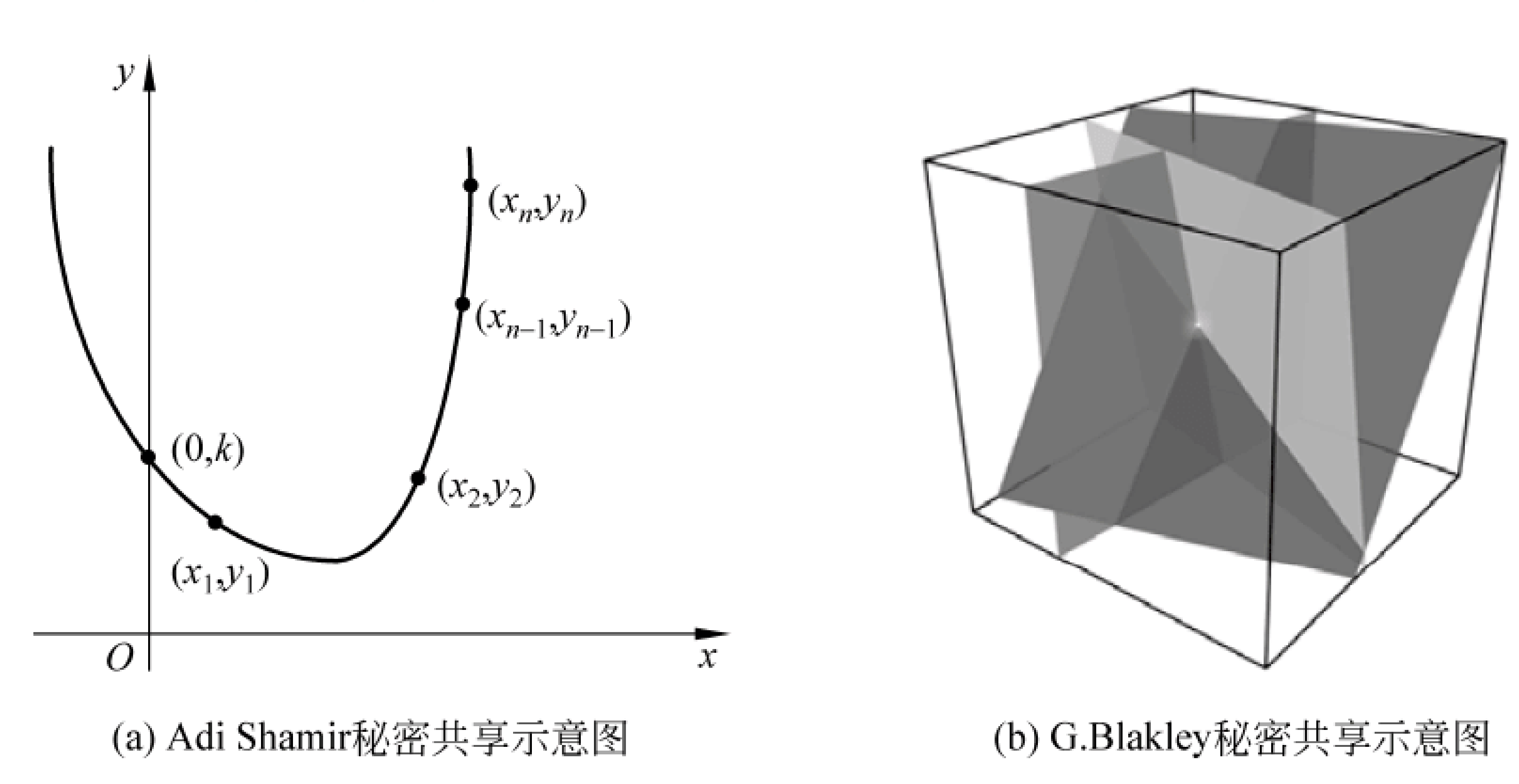
严格地说:秘密共享技术的基本要求是将秘密 k 分成 n 个份额 $s_1,s_2,\cdots,s_n$,使得以下条件满足:

(1) 已知任意 t 个 s_i,易于求出 k。

(2) 已知任意 $t-1$ 个 s_i 或更少的 s_i,不能确定 k。

因此,这种秘密共享也称为 (t,n) 门限(threshold)方案($t\leqslant n$)。

秘密共享的概念是 Adi Shamir 和 G. Blakley 于 1979 年分别独自给出的,Shamir 提出的方案是根据 Lagrange 插值公式构造了 (t,n) 门限方案,即为了重构 $t-1$ 次多项式,该多项式有 t 个未知的多项式系数,需要知道 t 个点,才能重构多项式曲线。如果只知道 $t-1$ 个点,则完全无法确定多项式系数。Blakley 的方案是利用线性几何投影法进行构造,其方法也很容易理解:一个三维空间的点可由 3 个二维平面的交点决定。一般地,若将共享的秘密映射到 t 维空间中的一个点,且构造的每一个份额都是包含这个点的 $t-1$ 维超平面的方程,那么 t 个或 t 个以上的这种超平面的交点刚好确定这个点。两种方案的示意图见图 13.3(右图来自维基百科)。



(a) Adi Shamir秘密共享示意图　　(b) G.Blakley秘密共享示意图

图 13.3　秘密共享概念示意

秘密共享技术提供了密钥抗泄露的可靠性,可以与其他密码学技术融合在一起,提供密码学技术的健壮性(提高抗共谋、容错、容入侵等能力)。秘密共享构成了门限密码学(threshold cryptography)的基础。门限密码学是指将基本的密码体制分布于若干参与者中的技术。例如,普通签名的门限版本,即门限签名(threshold signature),使用某种秘密分享方法将签名私钥在一些参与者中分享,使得参与者的适当子集可以联合发布签名,而不合格的子集则无法产生有效签名。同样的还有门限加密等。

13.4.2 Shamir 门限方案

首先给出门限方案的严格定义。

定义 13.1　设 t 和 n 为正整数,$t\leqslant n$。一个 (t,n) 门限方案是一种在 n 个参与者中分

享一个密钥 k 的方法，使得任意 t 个参与者在给出他们的秘密份额后可以恢复密钥 k，而任意 $t-1$ 个参与者在给出他们的秘密份额后不能恢复秘密（密钥）k。

设 (t,n) 门限方案中的 n 个参与者为 $P_1,P_2,\cdots,P_n$。密钥 k 由一个秘密分发者 D 选取，$D\notin\{P_1,P_2,\cdots,P_n\}$。$D$ 想让 $P_1,P_2,\cdots,P_n$ 分享密钥 k。D 秘密地分配给每人一个关于密钥 k 的份额。所谓份额就是密钥 k 的部分信息。当 $P_1,P_2,\cdots,P_n$ 中的任意 t 个人给出他们的份额后可以重建密钥 k，而任意 $t-1$ 个参与者在给出他们的份额后不能重建密钥 k。

一般的秘密共享体制包括两个协议：

(1) 秘密分发协议。在这个协议中，秘密分发者 D 在 n 个参与者中分享秘密 k，每个参与者 P 获得一个份额 s_i。

(2) 秘密重构协议。在这个协议中，任意不少于 t 个参与者一起合作，以自己的份额为输入，重构原秘密 k。

下面举两个特例以便于理解门限方案的概念。

(1) $(1,n)$ 门限方案（即 $t=1$）。

秘密分发协议：将秘密 k 分发给所有的参与者，每个参与者均得到一个 k，即 $s_i=k$，$(1\leqslant i\leqslant n)$。

秘密重构协议：可忽略。

该特例表明，只要有 1 个参与者，即可重构秘密。该方案退化为密钥备份方案，没有提高抵御密钥泄露的能力。

(2) (n,n) 门限方案（即 $t=n$）。

秘密分发协议：随机选取 s_i 分发 $P_i(1\leqslant i\leqslant n-1)$，将秘密 $s_n=k\bigoplus_{i=1}^{n-1}s_i$ 分发给参与者 s_n。

秘密重构协议：所有参与者的份额异或得到 k，即 $k=\bigoplus_{i=1}^{n}s_i$。

该特例表明，必须所有参与者参与才能重构，否则无法了解秘密信息。该方案能抵御密钥的泄露，除非所有份额都泄露了，才会暴露秘密 k。但该方案中的份额不能丢失，否则无法重构秘密。这一方案也叫做秘密分割(secret splitting)。

在初步认识了门限方案后，下面介绍 Shamir 的 (t,n) 门限方案。

Shamir 的 (t,n) 门限方案的秘密分发协议描述如下：

设 p 是一个素数，$p\geqslant n+1$。

(1) D 从 $\mathbb{Z}_p$ 中选取 n 个不同的非零元 $x_1,x_2,\cdots,x_n$，$n\leqslant p-1$。D 将 x_i 分配给参与者 P_i，$1\leqslant i\leqslant n$。$x_1,x_2,\cdots,x_n$ 可以公开。

(2) 如果 D 要让参与者 $P_1,P_2,\cdots,P_w$ 分享一个密钥 $k\in\mathbb{Z}_p$，则 D 秘密地随机选取 $t-1$ 个元素 $a_1,a_2,\cdots,a_{t-1}\in\mathbb{Z}_p$。

(3) 对 $1\leqslant i\leqslant n$，D 计算 $y_i=a(x_i)$，其中

$$a(x)=\left(k+\sum_{j=1}^{t-1}a_jx^j\right) \bmod p$$

(4) D 将 y_i 秘密地分配给 P_i，$1\leqslant i\leqslant n$。

Shamir 的 (t,n) 门限方案的基本思想是：D 构造了一个次数至多为 $t-1$ 的随机多项

式 $a(x)$，其常数项为密钥 k。每个参与者 P_i 得到了多项式 $a(x)$ 所确定的曲线上的一个点 (x_i, y_i)，$1 \leqslant i \leqslant n$。只有当点的个数有 t 个时，才能重构多项式 $a(x)$，从而得到秘密。注意，这里所有的运算都是有限域 $\mathbb{Z}_p$ 上的运算。

Shamir 的 (t,n) 门限方案的秘密重构协议如下：

假设 t 个参与者 $P_{i_1}, P_{i_2}, \cdots, P_{i_t}$ 想要重建密钥 k，则他们都给出自己的 x_{i_j} 和 y_{i_j}，$1 \leqslant j \leqslant t$。由于 $y_{i_j} = a(x_{i_j})$，所以可得到 t 个关于未知数 $k, a_1, a_2, \cdots, a_{t-1}$ 的线性方程：

$$\begin{cases} k + a_1 x_{i_1} + a_2 x_{i_1}^2 + \cdots + a_{t-1} x_{i_1}^{t-1} = y_{i_1} \\ k + a_1 x_{i_2} + a_2 x_{i_2}^2 + \cdots + a_{t-1} x_{i_2}^{t-1} = y_{i_2} \\ \qquad \vdots \\ k + a_1 x_{i_n} + a_2 x_{i_n}^2 + \cdots + a_{t-1} x_{i_n}^{t-1} = y_{i_n} \end{cases} \tag{13.1}$$

上述线性方程组可以写成矩阵的形式：

$$\begin{pmatrix} 1 & x_{i_1} & x_{i_1}^2 & \cdots & x_{i_1}^{t-1} \\ 1 & x_{i_2} & x_{i_2}^2 & \cdots & x_{i_2}^{t-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{i_t} & x_{i_t}^2 & \cdots & x_{i_t}^{t-1} \end{pmatrix} \begin{pmatrix} k \\ a_1 \\ \vdots \\ a_{t-1} \end{pmatrix} = \begin{pmatrix} y_{i_1} \\ y_{i_2} \\ \vdots \\ y_{i_t} \end{pmatrix} \tag{13.2}$$

设系数矩阵为 $\mathbf{A}$。显然，$\mathbf{A}$ 是一个 Vandermonde 矩阵。系数矩阵 $\mathbf{A}$ 的行列式为

$$\det(\mathbf{A}) = \prod_{1 \leqslant j < m \leqslant t} (x_{i_m} - x_{i_j}) \bmod p$$

因为 $x_1, x_2, \cdots, x_w$ 互不相同，所以 $\det(\mathbf{A}) \neq 0$。因此，线性方程组(13.1)有唯一解。这说明任意 t 个参与者能够重建密钥 k。

如果 $t-1$ 个参与者 $P_{i_1}, P_{i_2}, \cdots, P_{i_{t-1}}$ 想要重建密钥 k，则他们都给出自己的 x_{i_j} 和 y_{i_j}，$1 \leqslant j \leqslant t-1$，可以得到 $t-1$ 个关于未知数 $k, a_1, a_2, \cdots, a_{t-1}$ 的线性方程。由这 $t-1$ 个线性方程无法确定 t 个未知数 $k, a_1, a_2, \cdots, a_{t-1}$ 的唯一解。当然，$P_{i_1}, P_{i_2}, \cdots, P_{i_{t-1}}$ 可以猜测密钥 k 的一个值。假设他们猜测 $k = y_0$。因为 $a(0) = k = y_0$，所以他们可以得到 t 个线性方程。

$$\begin{pmatrix} 1 & x_{i_1} & x_{i_1}^2 & \cdots & x_{i_1}^{t-1} \\ 1 & x_{i_2} & x_{i_2}^2 & \cdots & x_{i_2}^{t-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{i_t} & x_{i_t}^2 & \cdots & x_{i_t}^{t-1} \\ 1 & 0 & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} k \\ a_1 \\ \vdots \\ a_{t-2} \\ a_{t-1} \end{pmatrix} = \begin{pmatrix} y_{i_1} \\ y_{i_2} \\ \vdots \\ y_{i_{t-1}} \\ y_0 \end{pmatrix} \tag{13.3}$$

线性方程组(13.3)的系数矩阵也是一个 Vandermonde 矩阵，其行列式不为 0。因此，线性方程组(13.3)有唯一解。这就是说，根据 $P_{i_1}, P_{i_2}, \cdots, P_{i_{t-1}}$ 对密钥 k 的每一个猜测值 y_0，都可以得到唯一一个多项式 $a_{y_0}(x)$，使得 $y_{i_j} = a_{y_0} x_{ij}$，$1 \leqslant j \leqslant t-1$，并且 $y_0 = a_{y_0}(0)$。因此，任意 $t-1$ 个参与者得不到关于密钥 k 的任何信息。

例 13.6 回到本节开头的问题。某个银行有 3 位出纳，他们每天都要开启保险库。为防止每位出纳可能出现的监守自盗行为，银行规定至少有两位出纳在场才能开启保险库。

保险库门上的电子锁是按 Shamir 的(t,n)门限方案设计的。假设银行管理者设置了素数 $p=13$。另外，$n=3$，$t=2$。假设开启保险库门上的电子锁的密钥为 $k=9$。银行管理者秘密选取 $a_1=5$，得到多项式 $a(x)=k+a_1x=9+5x$。银行管理者再选取 $x_1=1$，$x_2=3$，$x_3=5$。银行管理者将上述所有参数事先设置到电子锁内部的微处理器中。

银行管理者计算

$$y_1 = a(x_1) = (9+5\times 1) \bmod 13 = 1$$
$$y_2 = a(x_2) = (9+5\times 3) \bmod 13 = 11$$
$$y_3 = a(x_3) = (9+5\times 5) \bmod 13 = 8$$

并将 y_1,y_2,y_3 分别秘密分发给 3 位出纳 P_1,P_2,P_3。

假设出纳 P_1 和 P_3 想要开启保险库，则他们分别将 $y_1=1$ 和 $y_3=8$ 输入电子锁。电子锁内部的微处理器将计算出密钥 k。因为

$$k+a_1x_1 = y_1$$
$$k+a_1x_3 = y_3$$

两式相减得 $a_1(x_1-x_3)=y_1-y_3$，得 $a_1=\dfrac{y_1-y_3}{x_1-x_3}$。因此，密钥为

$$\begin{aligned}k &= y_1-a_1x_1 = y_1-\frac{y_1-y_3}{x_1-x_3}x_1 = \frac{y_1(x_1-x_3)-(y_1-y_3)x_1}{x_1-x_3}\\ &= \frac{x_3}{x_3-x_1}y_1+\frac{x_1}{x_1-x_3}y_3 = \left(\frac{5}{5-1}\times 1+\frac{1}{1-5}\times 8\right) \bmod 13\\ &= (5\times 4^{-1}\times 1+1\times(-4)^{-1}\times 8) \bmod 13\\ &= (5\times 4^{-1}\times 1+1\times 9^{-1}\times 8) \bmod 13\\ &= (5\times 10\times 1+1\times 3\times 8) \bmod 13 = 9\end{aligned}$$

电子锁内部的微处理器判定计算出的密钥是正确的密钥，保险库的门打开。

其实，更一般地，可利用 Lagrange 插值公式重构(t,n)门限方案中的密钥。当 t 个参与者 $P_{i_1},P_{i_2},\cdots,P_{i_t}$ 想要重建密钥 k 时，可以利用 Lagrange 插值公式容易地求得密钥 k，从而避免求解线性方程组的计算。

所谓插值公式(一个来源于数值分析的概念)，即快速确定一条满足经过点(x_{i_1},y_{i_1})，(x_{i_2},y_{i_2})，$\cdots$，(x_{i_t},y_{i_t})的曲线方程，易知，经过 t 个点可以唯一地确定一个次数至多为 $t-1$ 的多项式。根据 Lagrange 插值公式

$$a(x) = \sum_{j=1}^{t} y_{i_j} \prod_{1\leqslant m\leqslant t, m\neq j} \frac{x-x_{i_m}}{x_{i_j}-x_{i_m}}$$

容易验证

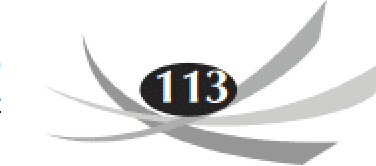
$$y_{i_j} = a(x_{i_j}),\quad 1\leqslant j\leqslant t$$

因为密钥 $k=a(0)$，所以

$$k = \sum_{j=1}^{t} y_{i_j} \prod_{1\leqslant m\leqslant t, m\neq j} \frac{x_{i_m}}{x_{i_m}-x_{i_j}}$$

令

$$b_j = \prod_{1\leqslant m\leqslant t, m\neq j} \frac{x_{i_m}}{x_{i_m}-x_{i_j}}$$



则直接给出秘密

$$k = \sum_{j=1}^{t} b_j y_{i_j}$$

例 13.7　设 $p=17, t=3, n=5, x_i=i, 1\leqslant i\leqslant 5$。假设 P_1, P_3, P_5 想要重构密钥 k，$y_1=8, y_3=10, y_5=11$。使用插值公式，直接计算得

$$b_1 = \frac{x_3 x_5}{(x_3-x_1)(x_5-x_1)} \bmod 17 = 3\times 5\times (3-1)^{-1}\times (5-1)^{-1} \bmod 17 = 4$$

$$b_2 = \frac{x_1 x_5}{(x_1-x_3)(x_5-x_3)} \bmod 17 = 1\times 5\times (1-3)^{-1}\times (5-3)^{-1} \bmod 17 = 3$$

$$b_3 = \frac{x_1 x_3}{(x_1-x_5)(x_3-x_5)} \bmod 17 = 1\times 3\times (1-5)^{-1}\times (3-5)^{-1} \bmod 17 = 11$$

从而密钥为 $k = \sum_{j=1}^{t} b_j y_{i_j} \bmod 17 = 4\times 8+3\times 10+11\times 11 \bmod 17 = 13$。

13.5　安全多方计算*

安全多方计算(secure multiparty computation)是指在一个互不信任的多用户网络中，各用户能够通过网络来协同完成可靠的计算任务，同时又能保持各自数据的安全性。安全多方计算其实是一种分布式协议，协议的目的是一组参与者希望共同计算某个约定的函数，每个参与者提供函数的一个秘密的输入。即 n 个成员 $p_1, p_2, \cdots, p_n$ 分别持有秘密的输入 $x_1, x_2, \cdots, x_n$，试图计算函数值 $y=f(x_1, x_2, \cdots, x_n)$，式中 f 是给定的函数。所谓“安全”是指既要保证函数值的正确性，又不暴露任何有关各自秘密输入的信息。如果存在安全可信第三方(TTP)，这个问题就容易解决，可让 TTP 计算出函数值，再将函数值公布给各参与者，但现实中很难找到这样的 TTP，于是安全多方计算协议便应运而生。

安全多方计算起源于图灵奖获得者姚期智(Andrew C. Yao)于 1982 年提出的百万富翁问题[①]。安全多方计算有很强的应用背景，如网上电子投标(拍卖)、网上商业谈判和电子选举计票等，还可以用来认证。安全多方计算目前已经成为密码学中一个极其重要的工具。

安全多方计算就是满足下述 3 个条件的密码协议：

(1) 多个参与者利用每个人的秘密输入来计算某个多变量复合函数的值。

(2) 参与者希望保持某种安全性(如机密性与正确性)，例如在安全电子投票协议中要保持投票者所投内容的机密性与票数计算的正确性。

(3) 协议既要保持在发生非协议参与者攻击行为下的安全性，也要保持在发生协议参与者攻击行为下的安全性，但不包括协议参与者的主动欺骗行为，即故意输入错误的秘密数据的情况。

平均薪水问题和百万富翁问题是安全多方计算协议的典型例子，下面以这两个协议为例介绍安全多方计算。

---

① A. C. Yao. Protocols for Secure Computation. Proc. of FOCS'82：160-164.

13.5.1 平均薪水问题

平均薪水问题是指：假设某公司的 n 个职员想了解他们每月的平均薪水有多少，但是每个职员又不想让任何其他人知道自己的薪水，那么他们的平均薪水如何来计算？不妨设公司有 n 个职员 $A_1,A_2,\cdots,A_n$，他们的薪水分别为 $x_1,x_2,\cdots,x_n$。平均薪水问题可以描述为：对 n 个秘密输入 $x_1,x_2,\cdots,x_n$，如何计算函数值

$$f(x_1,x_2,\cdots,x_n) = (x_1+x_2+\cdots+x_n)/n$$

1. 协议描述

为了在不需要任何第三方参与的情况下计算 $f(x_1,x_2,\cdots,x_n)$，可以执行如下的协议完成上述任务：

(1) n 个职员共同确定一种公钥加密体制（E 为加密算法，D 为解密算法），然后每个职员各自选定自己的公私钥对，不妨设职员 A_i 的公私钥对为$(\mathrm{pk}_i,\mathrm{sk}_i)$。

(2) A_1 选择一个随机数 r 并加到自己的薪水上得 $r+x_1$，然后把 $E_{\mathrm{pk}_2}(r+x_1)$发送给 A_2。

(3) A_2 解密得 $E_{\mathrm{sk}_2}(E_{\mathrm{pk}_2}(r+x_1))=r+x_1$ 后，加上自己的薪水得 $r+x_1+x_2$，然后把结果 $E_{\mathrm{pk}_2}(r+x_1+x_2)$发送给 A_3。

(4) A_3 执行与 A_2 类似的操作，$A_4,A_5,\cdots,A_{n-1}$继续同样的操作。

(5) A_n 用其私钥解密得 $E_{\mathrm{sk}_n}(E_{\mathrm{pk}_n}(r+x_1+x_2+\cdots+x_{n-1}))=r+x_1+x_2+\cdots+x_{n-1}$后，加上自己的薪水，然后把 $E_{\mathrm{pk}_1}(r+x_1+x_2+\cdots+x_n)$发送给 A_1。

(6) A_1 解密得 $E_{\mathrm{sk}_1}(E_{\mathrm{pk}_1}(r+x_1+x_2+\cdots+x_{n-1}))=r+x_1+x_2+\cdots+x_{n-1}$后，将其减去随机数 r，再除以总人数便得公司职员的平均薪水：

$$(x_1+x_2+\cdots+x_n)/n$$

A_1 向 $A_2,A_3,\cdots,A_n$ 公布平均薪水的结果。

例 13.8 设有 3 个职员 A_1,A_2,A_3，他们的薪水分别是 $x_1=2,x_2=6,x_3=7$，下面用上述协议求他们的平均工资。

不妨设 3 人协商好用 ElGamal 公钥密码体制进行加解密，g 和 p 分别表示本原元和大素数，$\mathrm{pk}_1=6,\mathrm{sk}_1=10,g_1=13,p_1=19;\mathrm{pk}_2=6,\mathrm{sk}_2=5,g_2=2,p_2=13;\mathrm{pk}_3=3,\mathrm{sk}_3=8,g_3=2,p_3=11$。

A_1 选择一个随机数 $r=2$ 并加到自己的薪水上得 $r+x_1=2+2=4$，然后随机取数 $n_1=7$，计算

$$c_{11} = g_2^{n_1} \bmod p_2 = 2^7 \bmod 13 = 11$$

$$c_{12} = (r+x_1)(\mathrm{pk}_2)^{n_1} \bmod p_2 = 4\times(6)^7 \bmod 13 = 2$$

最后把密文对$(c_{11},c_{12})=(11,2)$发送给 A_2。

A_2 用其私钥解密：

$$\begin{aligned} r+x_1 &= c_{12}/c_{11}^{\mathrm{sk}_2} \bmod p_2 = 2/11^5 \bmod 13 = 2\times(11^5)^{-1} \bmod 13 \\ &= 2\times 7^{-1} \bmod 13 = 2\times 2 \bmod 13 = 4 \end{aligned}$$

加上 A_2 的薪水得 $r+x_1+x_2=4+6=10$，然后随机取数 $n_2=3$，计算

$$c_{21} = g_3^{n_2} \bmod p_3 = 2^3 \bmod 11 = 8$$

$$c_{22} = (r + x_1 + x_2)(\mathrm{pk}_3)^{n_2} \bmod p_3 = 10 \times (3)^3 \bmod 11 = 6$$

最后把密文对$(c_{21}, c_{22}) = (8, 6)$发送给$A_3$。

A_3用其私钥解密：

$$r + x_1 + x_2 = c_{22}/c_{21}^{\mathrm{sk}_3} \bmod p_3 = 6/(8^8) \bmod 11 = 6 \times (8^8)^{-1} \bmod 11$$
$$= 6 \times 5^{-1} \bmod 11 = 6 \times 9 \bmod 11 = 10$$

加上A_3的薪水得$r+x_1+x_2+x_3=10+7=17$，然后随机选取数$n_3=11$，计算

$$c_{31} = g_1^{n_3} \bmod p_1 = 13^{11} \bmod 19 = 2$$
$$c_{32} = (r + x_1 + x_2 + x_3)(\mathrm{pk}_1)^{n_3} \bmod p_1 = 17 \times (6)^{11} \bmod 19 = 4$$

最后把密文对$(c_{31}, c_{32}) = (2, 4)$发送给$A_1$。

A_1用其私钥解密：

$$r + x_1 + x_2 + x_3 = c_{32}/c_{31}^{\mathrm{sk}_1} \bmod p_1 = 4/(2^{10}) \bmod 19 = 4 \times (2^{10})^{-1} \bmod 19$$
$$= 4 \times 17^{-1} \bmod 19 = 4 \times 9 \bmod 19 = 17$$

从而得$r+x_1+x_2+x_3=17-r=17-2=15$，最终得平均薪水为

$$f(x_1, x_2, x_3) = (x_1 + x_2 + x_3)/3 = 15/3 = 5$$

A_1向A_2，A_3公布平均薪水为5。

2. 关于协议的几点注释

上述协议假定每个职员是诚实的，即每个职员加上去的是他们的真实薪水，如果某个职员谎报了薪水，则最后A_1计算出的平均薪水就是错误的。

A_1必须是诚实的，否则在A_1计算出正确的平均薪水后，可能向其他人公布一个错误的平均数。

A_1在协议的第(6)步减去任意一个数其他人也无从知晓，为了防止A_1这样做，可利用前面提到的比特承诺方案让A_1对他选择的随机数r作出承诺。

若在协议执行过程中A_1选择的随机数r发生泄露，那么所有职员的薪水就会泄露。该随机数必须是一个较大的随机数，以防止穷举搜索攻击。

13.5.2　百万富翁问题

百万富翁问题是姚期智于1982年提出的第一个两方安全计算问题，百万富翁问题是：两个百万富翁在街头相遇，他们想比较谁更富有，但又不想让对方了解自己的财富有多少。如果他们能找到一个双方都可以信任的第三方来做此事，则问题很容易解决。但如果其中有一位百万富翁除了自己外谁也不相信，则问题就变得复杂，那么如何在不借助任何第三方的情况下比较他们财富的大小？下面给出了一个解决百万富翁问题的协议。

1. 协议描述

不妨设A和B是两个百万富翁，A拥有的真实财富是i百万，B拥有的真实财富是j百万，该问题可以用数学表示为：对两个秘密输入i和j，判断函数值$f(i,j)=i-j\leqslant 0$还是$f(i,j)=i-j>0$。假定A和B两人拥有的财富数目没有超过N百万，即$1\leqslant i,j\leqslant N$。为了在不让任何第三方参与的情况下比较$i$和$j$的大小，又不向对方泄露各自的财富，则他们可以执行如下的协议：

(1) A和B共同协商一种公钥加密体制(E为加密算法，D为解密算法)，不妨设B的

公钥和私钥分别是 pk_B 和 sk_B。

(2) A 随机选择一个大随机数 x，用 B 的公钥加密得 $c=E_{pk_B}(x)$，然后将 $c-i$ 发送给 B。

(3) B 首先计算 N 个数：$y_u=D_{sk_B}(c-i+u)$，$u=1,2,\cdots,N$。然后随机选择一个大素数 p，再计算 N 个数：$z_u=y_u \bmod p$，$u=1,2,\cdots,N$。接着验证对于所有的 $0\leqslant a\neq b\leqslant N-1$，是否都满足 $|z_a-z_b|\geqslant 2$，若不满足，则重新选择大素数 p 并重新验证。最后，B 将以下的 $N+1$ 个数串发送给 A：$z_1,z_2,\cdots,z_j,z_{j+1}+1,z_{j+2}+1,\cdots,z_N+1$ 和 p。

(4) 设 A 收到 $N+1$ 个数串的第 i 个数是 z_i，若满足 $z_i=x \bmod p$，则结论是 $i\leqslant j$；否则 $i>j$。

(5) A 将最终的结论告诉 B。

2. 协议举例

设 A 和 B 两个百万富翁的财富都不超过 1000 万，即 $i,j<10$，其中 A 的财富是 900 万，B 的财富是 400 万，即 A 和 B 的秘密数分别为 $i=9$ 和 $j=4$。

(1) A 和 B 选定用 RSA 公钥算法对数据加密，双方约定 $n=221$，且 $pk_B=35$，$sk_B=11$。

(2) A 随机选择整数 $x=92$，计算 $c=E_{pk_B}(x)=E_{35}(92)=92^{35} \bmod 221=105$，然后把 $c-i=105-9=96$ 发送给 B。

(3) 对 $u=1,2,\cdots,10$，B 分别计算 $y_u=D_{sk_B}(c-i+u)=D_{11}(96+u)$：

$$y_1=D_{11}(96+1)=D_{11}(97)=97^{11} \bmod 221=193$$
$$y_2=D_{11}(96+2)=D_{11}(98)=98^{11} \bmod 221=106$$
$$y_3=D_{11}(96+3)=D_{11}(99)=99^{11} \bmod 221=44$$
$$y_4=D_{11}(96+4)=D_{11}(100)=100^{11} \bmod 221=94$$
$$y_5=D_{11}(96+5)=D_{11}(101)=101^{11} \bmod 221=186$$
$$y_6=D_{11}(96+6)=D_{11}(102)=102^{11} \bmod 221=136$$
$$y_7=D_{11}(96+7)=D_{11}(103)=103^{11} \bmod 221=103$$
$$y_8=D_{11}(96+8)=D_{11}(104)=104^{11} \bmod 221=195$$
$$y_9=D_{11}(96+9)=D_{11}(105)=105^{11} \bmod 221=92$$
$$y_{10}=D_{11}(96+10)=D_{11}(106)=106^{11} \bmod 221=98$$

取素数 $p=109$，计算 $z_u=y_u \bmod p=y_u \bmod 109$，$u=1,2,\cdots,10$，得

$$z_1=y_1 \bmod 109=193 \bmod 109=84$$
$$z_2=y_2 \bmod 109=106 \bmod 109=106$$
$$z_3=y_3 \bmod 109=44 \bmod 109=44$$
$$z_4=y_4 \bmod 109=94 \bmod 109=94$$
$$z_5=y_5 \bmod 109=186 \bmod 109=77$$
$$z_6=y_6 \bmod 109=136 \bmod 109=27$$
$$z_7=y_7 \bmod 109=103 \bmod 109=103$$
$$z_8=y_8 \bmod 109=195 \bmod 109=86$$

$$z_9 = y_9 \bmod 109 = 92 \bmod 109 = 92$$
$$z_{10} = y_{10} \bmod 109 = 98 \bmod 109 = 98$$

（4）B检验数列84,106,44,94,78,28,104,87,93,99是一个“好数列”，然后将以下11个数发送给A：84,106,44,94,78,28,104,87,93,99,109。

（5）A检查该数列中的第9个数是93，因为$93 \neq 92 \bmod 109$，所以$i>j$，即A比B更富有。

（6）A将结果告诉B。

3. 协议说明

（1）当且仅当$i \leqslant j$时，数列$z_1, z_2 \cdots, z_j, z_{j+1}+1, z_{j+2}+1, \cdots, z_N+1, p$中才存在数$z_i$，满足$z_i = x \bmod p$，否则该数列中任何数模$p$都不与$x$同余。

（2）要求z_u中的任何两个数z_a、z_b满足$|z_a - z_b| \geqslant 2$是为了保证B发送给A的$N+1$个数的数列$z_1, z_2, \cdots, z_j, z_{j+1}+1, z_{j+2}+1, \cdots, z_N+1, p$中任意两个数不同，一般称这样的数列为“好数列”。因为若数列中存在两个数$z_m = z_n$，$m<n$，则A可以判断出B的秘密数的大致范围为$m \leqslant j < n$。

（3）A比B先知晓了最终的结果，若A欺骗B告诉他相反的结论，则该协议是不公平的。为了增加公平性，B可以要求与A交换角色，即原来A执行的步骤现由B执行，而由B执行的步骤改由A执行。这样B也可以首先得出结论。

（4）协议无法判断$i=j$的情况，这是该协议的一个缺点。即只知道$i \leqslant j$，相等的情况无法判定。

（5）协议假定秘密数是正整数，而若是一般的实数，可以考虑两个实数的最大整数部分并对协议做相应的修改。

（6）协议只涉及两方的安全计算，可将上述协议推广到任意多方的安全计算。

（7）当两方的秘密数很大时，协议的计算量非常大：设待比较的两个数的长度（十进制表示的位数）为n，则数的范围是10^n，是输入规模的指数。其中，协议第（3）步中的解密次数和协议第（4）步中的检验次数都是10^n，协议第（3）步中的检验次数为$10^{2n}/2$。因此计算复杂性为输入规模的指数函数。如果输入规模为100，则计算复杂性为$O(10^{100})$，这样的计算量实际上是不可能实现的。因此这个协议对于比较两个较大的数不实用。

第 3 部分

小　结

本部分介绍了实体认证协议、身份识别协议、密钥协商协议以及高级协议。本部分的重点是基于口令的实体认证协议、基于挑战应答的密钥协商协议、比特承诺、秘密共享，难点是身份识别协议的一般解释、零知识证明、安全多方计算。本部分的知识点在下图给出了概括：

- 第 3 部分的主要知识点
  - 实体认证
    - 基于口令
      - 基于口令的认证协议
      - 基于散列链的认证协议
      - 认证的密钥交换协议
    - 基于挑战应答协议
      - 基于对称密码：一路单向，两路单向，两路双向，三路双向
      - 基于公钥密码：一路单向，两路单向，两路双向，三路双向，两路并行认证协议
      - 基于散列函数构造
  - 身份识别
    - Fiat-Shamir 协议：$x = r^2 \bmod n$　$y = r \cdot s^e \bmod n$
    - Feige-Fiat-Shamir 协议：Fiat-Shamir 协议的并行版本
    - Guillou-Quisquater 协议：$x = r^b \bmod n$　$y = ru^e \bmod n$
    - Schnorr 协议：$\text{Commit} = g^r \bmod p$　$\text{response} = r + s \cdot \text{challenge} \bmod q$
    - Okamoto 协议：$\text{Commit} = g_1^{r_1} g_2^{r_2} \bmod p$　$\text{response}_1 = r + s_1 \cdot \text{challenge} \bmod q$　$\text{response}_2 = r + s_2 \cdot \text{challenge} \bmod q$
  - 密钥协商
    - DH ⟶ STS ⟶ MTI　ECMQV
    - 会议密钥　Shamir 3 次传递
  - 比特承诺
    - 比特承诺函数的选定：屏蔽性和绑定性
    - 比特承诺方案：承诺阶段和打开阶段
    - 利用单向散列函数构造
      - 基于 Goldwasser-Micali 概率加密体制的比特承诺
      - Petersen 比特承诺
    - 利用对称密码算法
    - 电话投币
      - 利用安全散列函数抛硬币
      - 采用平方根投硬币
      - 利用二次剩余投币
  - 零知识证明
    - 洞穴的故事
    - 图同构
    - Hamilton 路径
  - 不经意传输
    - 二选一
      - 基于大整数分解的二选一 OT
      - 基于离散对数问题的二选一 OT
    - 多选一
      - 基于单向函数的多选一 OT
      - 基于离散对数的多选一 OT
    - n 选 k
  - 秘密共享　Shamir 门限方案
  - 安全多方计算
    - 平均薪水问题
    - 百万富翁问题

扩展阅读建议

按照时间顺序阅读身份识别协议的系列经典论文是有益的。

Fiat, A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. Crypto'86, 1987: 186-194.

U. Feige, A. Fiat, A. Shamir. Zero Knowledge Proofs of Identity. Journal of Cryptology, 1988, 1: 77-94.

L. Guillou, J. Quisquater. A Practicial Zero-Knowledge Protocol fitted to Security Microprocessors Minimizing Both Transmission and Memory. Eurocrypt 1988, 123-128.

Schnorr. Efficient Signarure Generation for Smart Cards. Journal of Cryptology, 1991, 4(3): 161-174.

T. Okamoto. Provable Secure and Practical Identificaiton Schemes and Corresponding Digital Signature Schemes. Crypto'92, 1992: 31-52.

秘密共享方面的经典文献如下：

A. Shamir. How to Share a Secret. Communication of the ACM 1979, 22(11): 612-613.

C. Asmuth J. Bloom. A Modular Approach to Key Safeguarding. IEEE Transaction of Information Theory 1983,29(2): 208-210.

P. Feldman. A Proactical Scheme for Non-Interactive Verifiable Secret Sharing. FOCS87, 427-437.

G.J. Simmons. An Introduction to Shared Secret and/or Shared Control Schemes and Their Applications. in Comtemporary Cryptography: The Science of Information Integrity. G. J. Simmons (Ed.)IEEE, 1992: 441-497.

M. Naor, A. Shamir. Visual Cryptogrphy. EuroCrypto94, 1995: 1-12.

刘木兰,张志芳.密钥共享体制和安全多方计算.北京：电子工业出版社,2000.

第 4 部分

基于身份的密码学和可证明安全性

第 14 章 基于身份的公钥密码学

14.1 概念、困难假设与 IBE

14.1.1 基于身份的公钥密码学概念的提出

前面曾经介绍，公钥证书的目的是将用户身份 ID 与用户的公钥进行绑定，绑定的可信性和权威性表现为 CA 为这一绑定进行了签名，即公钥证书。一个很直接的想法是能否避免这种绑定，即不需要公钥证书。1984 年，Shamir 提出了一种基于身份的加密(Identity-Based Encryption，IBE)的思想，直接将用户的身份作为公钥，这样就不需要绑定公钥和身份，于是不需要使用任何证书，以此来简化公钥基础设施(PKI)中基于证书的密钥管理过程。

基于身份的公钥加密可以用在加密电子邮件的场景：用户 A 给用户 B 发加密的电子邮件，B 的邮件地址是 bob@company. com，A 只要将"bob@company. com"作为 B 的公钥来加密邮件即可。当 Bob 收到加密的邮件后，与一个可信第三方即密钥生成中心(Key Generation Center，KGC)联系，与向 CA 证明自己身份一样，B 向 KGC 证明自己，并从服务器获得解密用的私钥，然后解密阅读邮件。该过程如图 14.1 所示。

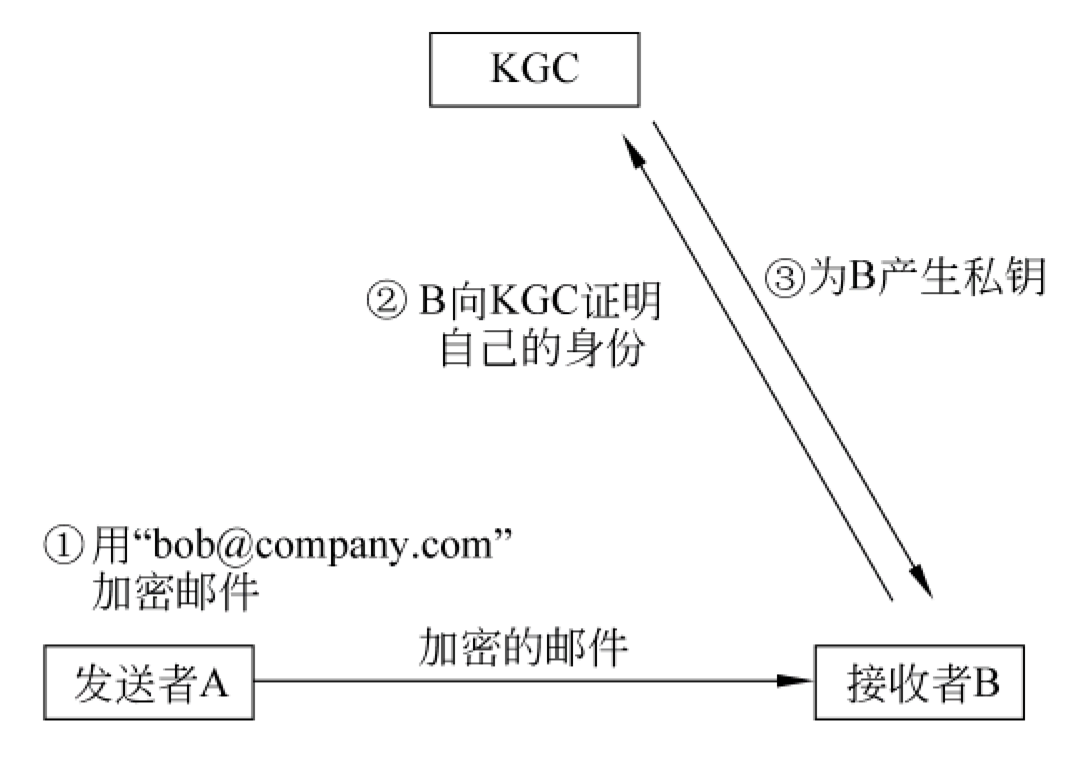


图 14.1 基于身份的加密方案示意图

与现有的安全电子邮件相比，即使 B 还未建立自己的公钥证书，A 也可以向 B 发送加密的邮件。因此这种方法避免了公钥密码体制中公钥证书从生成、签发、存储、维护、更新到撤销这一复杂的生命周期过程。

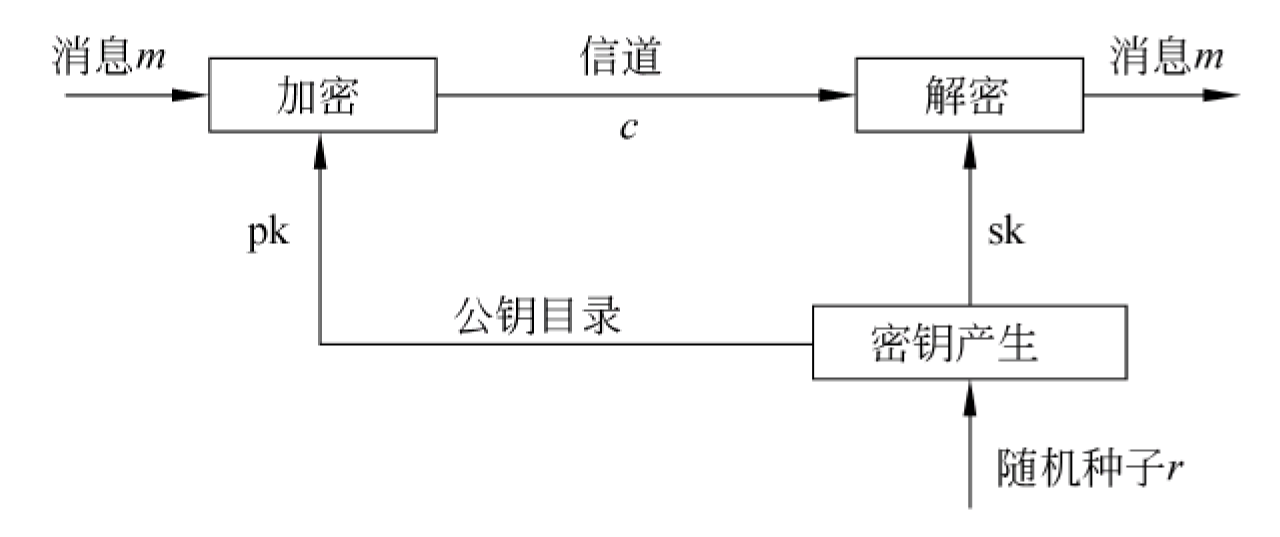
基于身份的密码系统与基于公钥证书(公钥基础设施)的密码系统相比的好处在于：

(1) 用户之间既不用交换私钥也不用交换公钥。

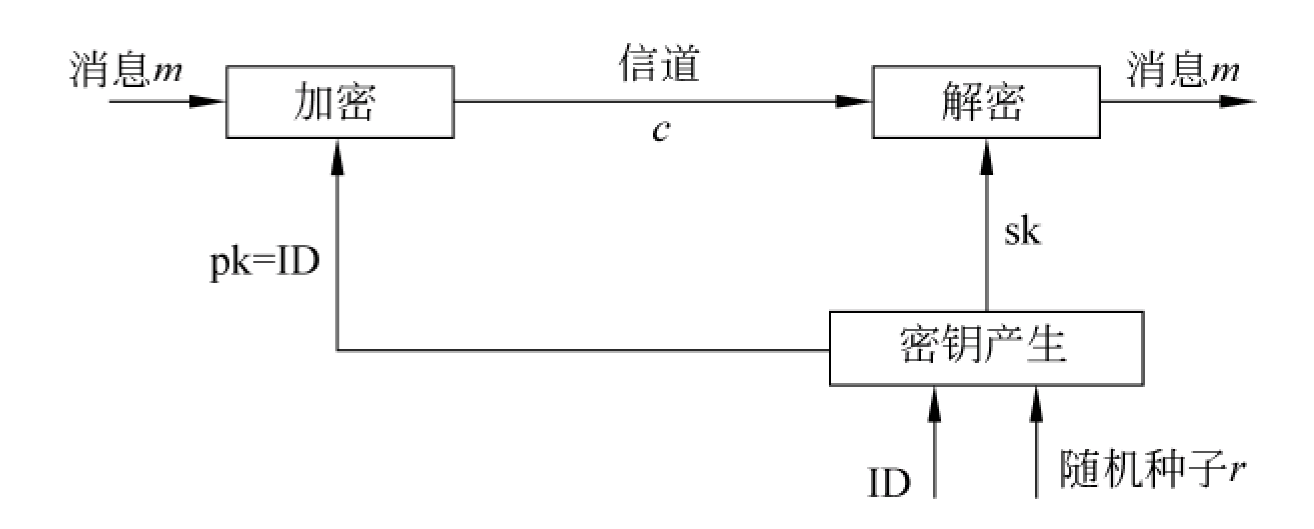
(2) 不需要公共的证书服务器。

(3) 只在系统建立阶段需要一个可信的机构为系统中每个用户生成密钥,且用户绝对信任该可信机构。

图 14.2 给出上述两种公钥密码体制的区别示意图。



(a) 基于证书的公钥密码体制



(b) 基于身份的公钥密码体制

图 14.2 基于公钥证书的公钥密码体制与基于身份的公钥密码体制的区别

这里的随机种子其实就是密钥生成中心(KGC)的主密钥,这要求满足两个附加条件:

(1) 当知道随机种子 r 时,能从 pk(即 ID)很容易求出 sk。

(2) 从一对特定的 sk 和 pk(即 ID)推测出种子 r 是困难的。

自 Shamir 提出这种 IBE 的新思想后,由于没有找到有效的实现工具,其实现方法一直是一个公开问题。直到 2001 年,Dan Boneh 和 Matt Franklin 提出了第一个实用的基于身份的公钥加密方案。他们的方案使用了椭圆曲线上的双线性映射(如 Weil 配对和 Tate 配对)。双线性映射(bilinear mapping)是满足条件 $\text{Pair}(aX,bY)=\text{Pair}(bX,aY)$ 的映射 Pair,其中 a 和 b 是整数,X 和 Y 是椭圆曲线上的点。用户的身份映射为一对公钥/私钥对。

目前,基于身份的密码技术也展开了标准化进程。2006 年国际标准化组织(ISO)出台了相关标准 ISO 4888—3。IEEE 也组织了专门的基于身份的密码学工作组(IEEE P1363.3),IETF 的 SMIME 工作组则进行了基于身份密码技术应用在电子邮件中的标准化工作。

思考 14.1 能否利用与 ElGamal 中的 $(g^r)^a=(g^a)^r$ 类似的结构构造基于 ID 的公钥加密方案。

回顾ElGamal公钥加密的构造思路：通过可交换的单向函数，临时选择一个随机数 r 构造一个类似于“一次一密”的对称密钥，然后使用该对称密钥加密。“一次一密”表现在每次加密都选取一个随机数构造用于加密的密钥。在基于身份的加密中，和 $(g^r)^a=(g^a)^r$ 类似的结构就是 $\mathrm{Pair}(aX,bY)=\mathrm{Pair}(bX,aY)$。根据这一思想，构造方案如下：

(1) 初始化。密钥生成中心(KGC)选取一条椭圆曲线、秘密整数 s 和椭圆曲线上的一点 P，公开 P 和 sP。

(2) 加密。发送方A想向接收方B发送消息 M，首先将B的身份(如bob@company.com)经散列函数映射到椭圆曲线上的一个点，记为 Q_{ID}，然后取一个秘密的随机数 r，计算 $k=\mathrm{Pair}(rQ_{\mathrm{ID}},sP)$，作为加密密钥。最后将加密结果 $E_k(M)$ 和 rP 发给接收方B。其中 E 是对称密钥加密算法。

(3) 私钥产生。接收方B收到 $E_k(M)$ 和 rP 后，向密钥服务器提出申请，服务器在对B认证后，计算 sQ_{ID} 并发送给B，B以 sQ_{ID} 作为私钥。

(4) 解密。B收到私钥后，计算 $k=\mathrm{Pair}(sQ_{\mathrm{ID}},rP)$，使用 k 及 E 对密文解密。由于映射Pair的性质，B计算的 k 与A使用的 k 相等。其他人不知道私钥 sQ_{ID}，故无法得到 k。

容易看到，方案的关键就是 $k=\mathrm{Pair}(rQ_{\mathrm{ID}},sP)=\mathrm{Pair}(sQ_{\mathrm{ID}},rP)$。

14.1.2　双线性映射和双线性DH假设

前面多次提到，公钥密码学加密(或签名)方案都是基于某个困难性假设来构造。本节介绍一个新的安全性假设——双线性DH假设。用 $\mathbb{Z}_p$ 代表在模 q 加法群 $\{0,1,\cdots,q-1\}$。对于阶为素数的群 G，用 G^* 代表集合 $G/\{O\}$，这里 O 为 G 中的单位元素。用 $\mathbb{Z}^+$ 代表正整数集。

1. 双线性映射

设 q 是一个大素数，G_1 和 G_2 是两个阶为 q 的群，其上的运算分别称为加法和乘法。G_1 到 G_2 的双线性映射 $e:G_1\times G_1\rightarrow G_2$ 满足下面的性质：

(1)双线性：如果对任意 $P,Q,R\in G_1$ 和 $a,b\in\mathbb{Z}$，有 $e(aP,bQ)=e(P,Q)^{ab}$ 或 $e(P+Q,R)=e(P,R)\cdot e(Q,R)$ 和 $e(P,Q+R)=e(P,Q)\cdot e(P,R)$，那么就称该映射为双线性映射。

(2) 非退化性：映射不把 $G_1\times G_2$ 中的所有元素对(即序偶)映射到 G_2 中的单位元。由于 G_1,G_2 都是阶为素数的群，这意味着：如果 P 是 G_1 的生成元，那么 $e(P,P)$ 就是 G_2 的生成元。

(3) 可计算性：对任意的 $P,Q\in G_1$，存在一个有效算法计算 $e(P,Q)$。

Weil配对和Tate配对是满足上述3条性质的双线性映射。

2. DDH问题

相应地，可以给出群 G_1 中的判定性Diffie-Hellman问题，简称DDH(Decision Diffie-Hellman)问题：已知 P,aP,bP,cP，判定 $c=ab \bmod q$ 是否成立，其中 P 是 G_1^* 中的随机元素，a,b,c 是 $\mathbb{Z}_q^*$ 中的随机数。

由双线性映射的性质可知：$c=ab \bmod q \Leftrightarrow e(P,cP)=e(aP,bP)$，因此可将判定是否成立转变为判定 $c=ab \bmod q$ 是否成立，所以 G_1 中的DDH问题是简单的。

3. CDH 问题

G_1 中的计算性 Diffie-Hellman 问题简称 CDH(Computational Diffie-Hellman)问题，是指已知 P, aP, bP，求 abP，其中 P 是 G_1^* 中的随机元素，a, b 是 $\mathbb{Z}_q^*$ 中的随机数。与 G_1 中的 DDH 问题不同，G_1 中的 CDH 问题不因引入双线性映射而解决，因此它仍是困难问题。

4. BDH 问题和 BDH 假设

由于 G_1 中的 DDH 问题简单，那么就不能用它来构造 G_1 中的密码体制，IBE 体制的安全性是基于 CDH 问题的一种变形，称为双线性 DH 假设。

双线性 DH 问题简称为 BDH(Bilinear Diffie-Hellman)问题，是指给定$(P, aP, bP, cP)(a, b, c \in \mathbb{Z}_q^*)$，计算 $w = e(P, P)^{abc} \in G_2$，其中 e 是一个双线性映射，P 是 G_1 的生成元，G_1, G_2 是阶为素数 q 的两个群。目前还没有有效的算法解决 BDH 问题，因此可假设 BDH 问题是一个困难问题，这一假设即为 BDH 假设。

14.1.3 Boneh Franklin IBE 方案

1. IBE 方案的一般定义

基于身份的加密方案通常由 4 个算法组成：系统参数建立，密钥产生，加密过程，解密过程。系统中包含一个可信第三方 KGC。

(1) 系统参数建立：输入一个安全参数，输出系统参数和主密钥(mk)。系统参数包括明文空间 M、密文空间 C、散列函数等，完全公开。主密钥用于计算私钥，由 KGC 秘密保存。

(2) 密钥产生：输入系统参数、主密钥和用作公钥的任意字符串 $\text{ID} \in \{0,1\}^*$，输出与公钥 ID 对应的私钥 d_{ID}。

(3) 加密过程：输入系统参数、公钥 ID 和明文 $m \in M$，返回相应的密文。

(4) 解密过程：输入系统参数、私钥 d_{ID} 和密文 $c \in C$，返回相应的明文。

2. Boneh-Franklin IBE 方案

前面已经提到，2001 年 Dan Boneh 和 Matt Franklin 提出了第一个实用的基于身份的公钥加密方案。具体方案如下：令 k 是安全参数，g 是 BDH 参数生成算法，其输出包括素数 q，两个阶为 q 的群 G_1 和 G_2，一个双线性映射 $e: G_1 \times G_1 \to G_2$ 的描述。k 用来确定 q 的大小，例如可以取 q 为 k 比特长。

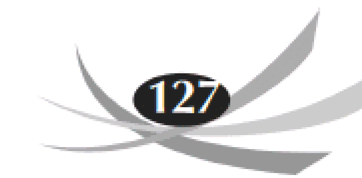
(1) 初始化。给定安全参数 $k \in \mathbb{Z}^+$，算法运行如下：

① 输入 k 后运行 g，产生素数 q，两个阶为 q 的群 G_1, G_2，一个双线性映射 $e: G_1 \times G_1 \to G_2$。选择一个随机生成元 $P \in G_1$。

② 随机选取一个 $s \in \mathbb{Z}_q^*$，确定 $P_{\text{pub}} = sP$。

③ 选取一个散列函数 $H_1: \{0,1\}^* \to G_1^*$。对某个 n，再选一个散列函数 $H_2: G_2 \to \{0,1\}^n$。(这里为了能够证明方案的安全性，需要把 H_1, H_2 视为随机预言模型(random oracle model)。随机预言模型具有一个性质：对任一输入，其输出的概率分布与均匀分布在计算上是不可区分的。)

消息空间为 $M = \{0,1\}^n$，密文空间为 $C = G_1^* \times \{0,1\}^n$。系统参数为 param$\langle q, G_1,$



$G_2, e, n, P_{pub}, H_1, H_2\rangle$是公开的。$s$ 为主密钥，是保密的。

（2）加密。用接收方的身份 ID 作为公钥加密消息 $m \in M$，有 3 步：

① 计算 $Q_{ID} = H_1(ID) \in G_1^*$。

② 选择一个随机数 $r \in \mathbb{Z}_q^*$。

③ 确定密文 $c = \langle rP, m \oplus H_2(g_{ID}^r)\rangle$，这里 $g_{ID} = e(Q_{ID}, P_{pub}) \in G_2^*$，$\oplus$是异或运算。

（3）密钥产生。对于一个给定的比特串 $ID \in \{0,1\}^*$，首先计算 $Q_{ID} = H_1(ID) \in G_1^*$，然后确定秘密密钥 $d_{ID} = sQ_{ID}$，其中 s 为主密钥。

（4）解密。设密文为 $c = \langle U, V\rangle \in C$，用秘密密钥 d_{ID} 计算 $V \oplus H_2(e(d_{ID}, U)) = m$。

解密成功是因为 $e(d_{ID}, U) = e(sQ_{ID}, rP) = e(Q_{ID}, P)^{sr} = e(Q_{ID}, P_{pub})^r = g_{ID}^r$。

思考 14.2　Boneh-Franklin IBE 的设计机理什么？

很明显，该加密构造的思想与 ElGamal 加密是一样的。g_{ID}^r是实质上的临时共享加密密钥，相当于 ElGamal 中的 y^r，rP 是构造共享加密密钥的一个“种子”，相当于 ElGamal 加密中的 g^r。接收者利用自己的私钥和收到的“种子”rP，通过 $e(d_{ID}, rP) = g_{ID}^r$构造了临时的共享加密密钥 g_{ID}^r。

14.2　基于身份的密钥共享体制

14.2.1　SOK 密钥共享体制

利用配对的良好结构，自然可以像 DH 方法那样构造密钥共享体制。2000 年 Sakai、Ohgishi 及 Kasahara 提出了基于身份的密钥共享体制（SOK 密钥共享体制），像基于身份的加密一样，需要一个可信的权威机构（KGC）来建立系统参数。

1. SOK 密钥共享体制描述

SOK 密钥共享体制由下列 3 部分组成：

（1）系统参数建立：由 KGC 完成，产生整个系统的参数及主密钥（mk）。

（2）用户密钥生成：由 KGC 完成，利用主密钥及用户传输的任意比特串 ID，产生对应的用户的私钥。

（3）密钥共享体制：用户完成。这一步骤由两个终端用户以非交互式方式完成。对每个终端用户，用自己的私钥以及预定的另一通信方的公钥（ID）产生这两个终端用户的共享密钥。

下面具体描述这 3 个部分的实现步骤：

（1）系统参数建立。

建立两个阶为 p 的群$(G_1, +)$与$(G_2, \cdot)$，以及改进的 Weil 配对：

$e: G_1 \times G_2 \to G_2$。任选 G_1 的一个生成元 $P \in G_1$。随机选取 $s: 1 \leqslant s \leqslant p-1$，并置 $P_{pub} = sP$（s 为主密钥）。选择一个密码意义上的单向散列函数 $h: \{0,1\}^* \to G_1$（该散列函数用于将用户的 ID 映射成 G_1 中的元素）。TA 将系统参数$(G_1, G_2, e, P, P_{pub}, h)$公开，而秘密保存主密钥 d 为系统私钥。

（2）用户密钥生成。



设 ID_A 表示用户 Alice 的唯一可识别的身份。且假定 ID_A 包含足够多的信息使得本系统中任何其他用户均不可能也以 ID_A 作为其身份。计算 $P_{ID_A}=h(ID_A)$，P_{ID_A} 作为 Alice 的基于身份 ID_A 的公钥。计算 sP_{ID_A} 并置 Alice 的私钥为 $d_{ID_A}=sP_{ID_A}$。

(3) 共享密钥生成。

对用户 Alice 与 Bob，ID_A 与 ID_B 分别表示他们互相已知的身份信息。因而 P_{ID_A} 与 P_{ID_B} 就是分别表示他们的互相已知的公钥。且假定 ID 包含足够多的信息使得本系统中任何两个用户有不同的 ID 作为其身份。

Alice 通过计算 $K_{AB}=e(d_{ID_A},P_{ID_B})$ 得到一个共享密钥 $K_{AB}\in G_2$。

Bob 通过计算 $K_{BA}=e(d_{ID_B},P_{ID_A})$ 得到一个共享密钥 $K_{BA}\in G_2$。

K_{AB} 与 K_{BA} 是相等的：这是因为

$$K_{AB}=e(d_{ID_A},P_{ID_B})=e(sP_{ID_A},P_{ID_B})=e(P_{ID_A},P_{ID_B})^s$$

$$K_{BA}=e(d_{ID_B},P_{ID_A})=e(P_{ID_B},sP_{ID_A})=e(P_{ID_B},P_{ID_A})^s$$

由 e 的对称性可知 $K_{AB}=K_{BA}$。从该体制可看出，Alice 与 Bob 没有进行交互通信就获得了一个共享密钥 K_{AB}。有时候也称为非交互式密钥共享(non-interactive key agreement)。这也就是基于身份的公钥体制也被称为非交互公钥体制(non-interactive public key cryptography)的原因。

同时，容易思考到，这一方式与 DH 密钥交换的基本思想十分类似。

2. 安全性讨论

系统主密钥 s 的安全性由求解群 G_1 上的离散对数问题的困难性保证。这是因为非 TA 者要想知道 s，就必须求解群 G_1 上的离散对数问题，已知 P 与 P_{pub}，求 s 使得 $P_{pub}=sP$。

Alice 的私钥 d_{ID_A} 的安全性是群 G_1 上的 CDH 问题的困难性保证。这是因为：由 $P_{ID_A}\in G_1$ 知 P_{ID_A} 可由 P 生成，于是存在正整数 $a(0\leqslant a\leqslant p-1)$ 使 $P_{ID_A}=aP$，从而

$$d_{ID_A}=sP_{ID_A}=s(aP)=(sa)P$$

这表明，已知 sP(即 P_{pub})及 aP(即 P_{ID_A})，要找 d_{ID_A}，即需计算 $(sa)P$。这显然是一个 CDH 问题。

如果除 Alice、Bob 及 TA 外的任何攻击者想要通过公开信息($P,P_{ID_A},P_{ID_B},P_{pub}$)获取 K_{AB}，则他必须解一个本质上为椭圆曲线上的 CDH 问题。因此，SOK 密钥共享体制在 CDH 问题困难性的假定下是安全的。

14.2.2 基于配对的三方 DH 密钥协商协议

将 14.2.1 节的方案进一步扩展，思考一个问题：如何建立三方同享的通信密钥？

2000 年 A. Joux 利用配对技术设计出了一种非常简单的三方密钥协议。Joux 称该协议为“三方 Diffie-Hellman”协议。2003 年 Wenbo Mao 利用经修改的 Weil 配对 e 简化了 Joux 的协议。称简化后的协议为 Joux-Mao 三方 Diffie-Hellman 协议。

Joux-Mao 三方 Diffie-Hellman 协议描述如下。设 Alice、Bob 及 Charlie 要建立他们之间的一个共享密钥。实现步骤如下：

(1) 公开参数建立：某有限域 F_q 上的一条超奇异椭圆曲线 $E(E_q)$；p 是一个素数，$p|$

$\#E(E_q)$；l 是正整数；e 是 $E(F_{q^l})$上的经变形映射修改的 Weil 配对；P 是 $E(F_{q^l})$中阶为 p 的点，G_1 是由 P 生成的 p 阶子群。

(2) 各方密钥建立：Alice、Bob 及 Charlie 分别秘密选择大于 1 小于 p 的整数 a、b、c 作为各自的私钥，并各自计算自己的公钥

$$P_A = aP,\quad P_B = bP,\quad P_C = cP$$

(3) 三方互相交换他们的公钥，即在公共信道上公布他们各自的公钥。

(4) 计算三方的共享密钥：Alice、Bob 及 Charlie 各自计算 $e(P_B, P_C)^a$、$e(P_C, P_A)^b$ 及 $e(P_A, P_B)^c$。

因为

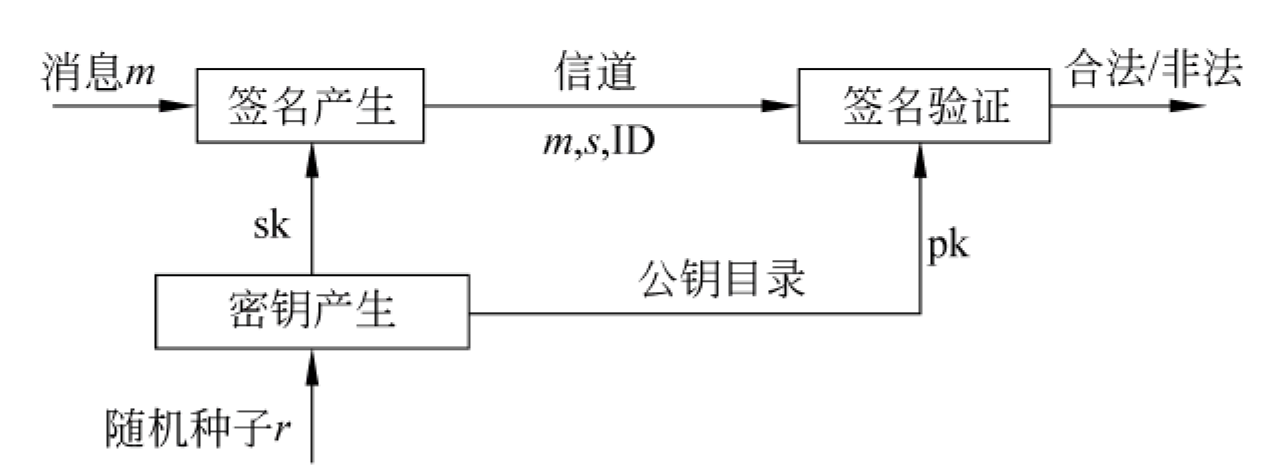
$$e(P_B, P_C)^a = e(P_C, P_A)^b = e(P_A, P_B)^c = e(P, P)^{abc}$$

所以 Alice、Bob 及 Charlie 建立了他们的共享密钥 $e(P, P)^{abc}$。

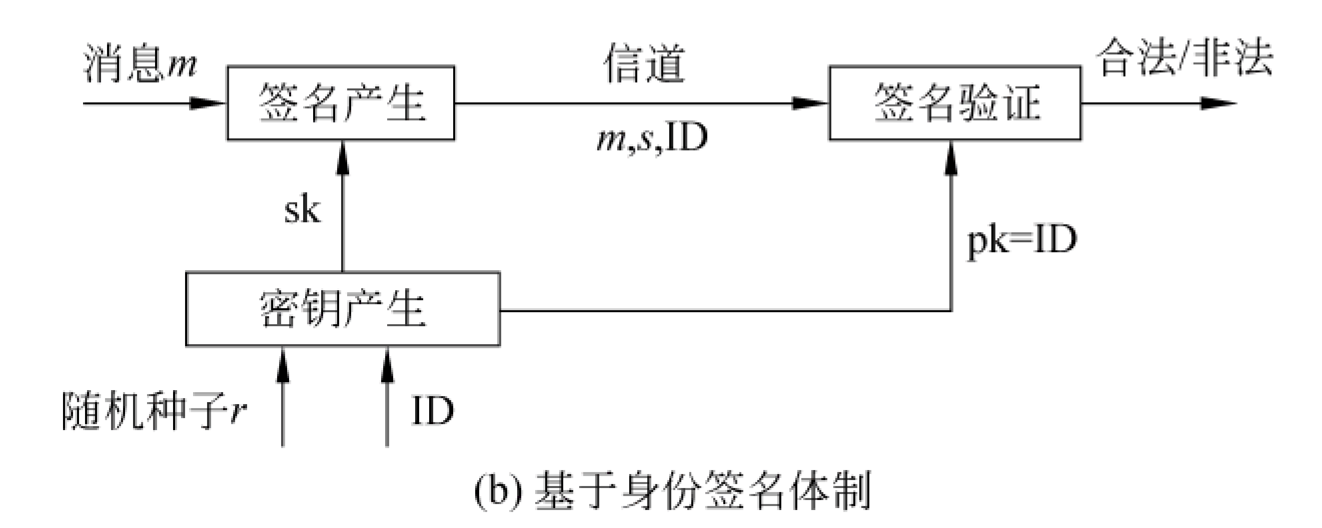
由于该三方 Diffie-Hellman 协议与原两方 Diffie-Hellman 密钥协商协议一样没有进行身份认证，所以易受到中间人的攻击。

14.3　基于身份的签名

基于身份的签名(Identity-Based Signature，IBS)可以通过双线性对构造，如 Cha-Cheon IBS；也可以不通过双线性对构造，如 Shamir IBS。基于公钥证书的签名体制与基于身份的签名体制区别在图 14.3 中给出。



(a) 基于公钥证书的签名体制



(b) 基于身份签名体制

图 14.3　两种签名体制的区别

14.3.1　Shamir 基于身份的签名

Shamir 提出的基于身份(ID)的数字签名体制由下列 4 个步骤组成：

(1) 参数建立(setup):由 KGC 完成。产生系统参数及主密钥(mk)。

(2) 用户私钥生成(user key generation):由 KGC 完成。利用 mk 及用户传输的任意比特串 ID,产生的该 ID 用户的私钥 $d_{\mathrm{ID_A}}$。

(3) 签名(sign):由签名者完成。对输入的信息 m,用签名者的私钥 $d_{\mathrm{ID_A}}$ 对信息 m 进行签名得到 s。

(4) 验证(verify):由验证者完成。对输入的信息 m 及相应的签名 s,利用 ID 验证签名的真实性。

图 14.4 给出了基于身份的签名的示意图。

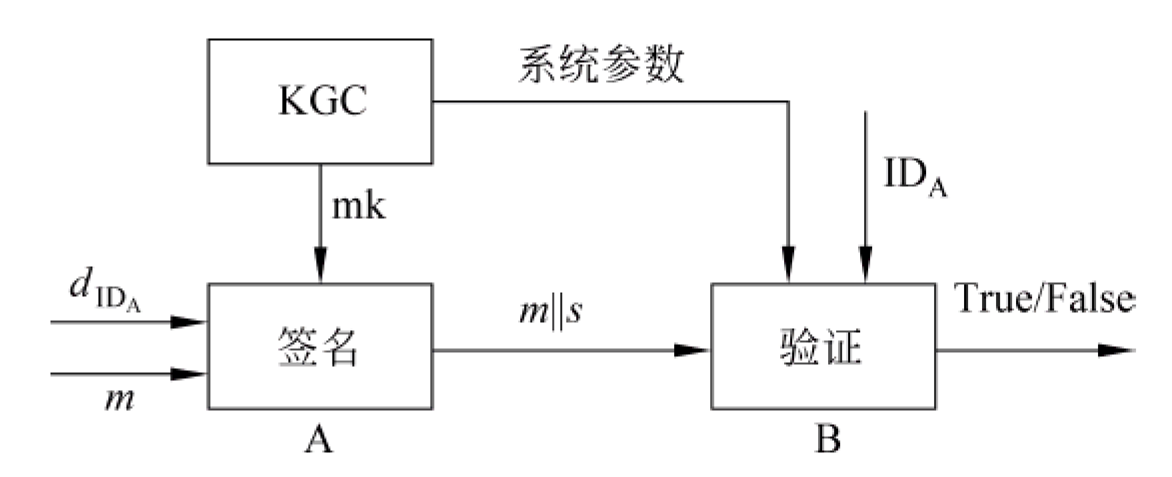


图 14.4 基于身份的数字签名产生和验证

Shamir 的基于身份的数字签名体制的算法描述如下:

(1) 系统参数建立。

N:秘密选择两个大素数 p,q,计算其积 $N=pq$。

e:选择正整数 e 满足 $\gcd(e,\phi(N))=1$(这里(N,e)为系统中用户使用的公开参数)。

d:计算出满足条件 $ed=1(\bmod\ \phi(N))$的正整数 d(d 为 KGC 的主密钥)。

h:$\{0,1\}^*\to\mathbb{Z}_{\phi(N)}$($h$ 是一个密码学意义上的单向散列函数)。

KGC 保存 d 作为系统的私钥,即主密钥,而公开系统参数为(N,e,h)。

(2) 用户私钥的生成。

设 ID 表示用户 Alice 的唯一可识别的身份。在对 Alice 完成了物理意义上的身份识别并确定了 ID 的唯一性后,KGC 计算

$$g=\mathrm{ID}^d(\bmod\ N)$$

将计算结果值作为 Alice 的私钥发给 Alice。

(3) 签名生成。

设 Alice 要对信息 $m\in\{0,1\}^*$ 进行签名。Alice 随机选择一个数 $r\in\mathbb{Z}_N^*$,并计算

$$t=r^e(\bmod\ N)$$

$$s=g\cdot r^{h(t\|m)}(\bmod\ N)$$

数对(s,t)就是 Alice 生成的签名。

(4) 签名验证。

设有信息 m 及其签名(s,t)。Bob 利用 Alice 的身份 ID 验证她的签名。如果 $s^e=\mathrm{ID}\cdot t^{h(t\|m)}(\bmod\ N)$,则 m 的签名(s,t)为真。

Shamir 的基于身份的签名算法是根据 RSA 签名来构造的,由此可以看出,在 Bob 验证 Alice 的签名时,只需要经 TA 核实的 ID。而在基于公钥证书基础设施的签名算法中,Bob 在验证 Alice 的签名前,必须先获得 Alice 的公钥证书,并验证 Alice 的公钥证书。这

一优点使得在基于身份的数字签名体制中,签名者不需要将公钥证书传输给验证者,这样可节省通信带宽。

观察签名验证方程:$s^e = \mathrm{ID} \cdot t^{h(t \| m)} \pmod N$。当 Bob 验证 Alice 的签名时,表明 Alice 拥有 $\mathrm{ID} \cdot t^{h(t \| m)}$ 及其模 N 的唯一 e 次根,也就是 s。这个唯一性是由 $\gcd(e, \phi(N)) = 1$ 保证的。

任何人要构造 $\mathrm{ID} \cdot t^{h(t \| m)}$ 并不困难。例如,可以选一个随机数 t,计算 $h(t \| m)$,再计算 $t^{h(t \| m)} \pmod N$,最后乘以 ID 即可。但是,想求出与该值对应的模 N 的唯一的 e 次根是很困难的,等同于已知 a 求方程 $x^e = a \pmod N$ 的解 x,即等同于求解 RSA 问题;另外,窜扰者想修改 t,即找出两个不同的 t 与 t_1 使 $\mathrm{ID} \cdot t^{h(t \| m)} = \mathrm{ID} \cdot t_1^{h(t_1 \| m)}$ 或 $t^{h(t \| m)} = t_1^{h(t_1 \| m)}$ 是困难的。

14.3.2 Cha-Cheon 基于身份的签名

Cha-Cheon 方案的构造与 Shamir 基于身份的签名十分相似,区别是使用了双线性映射。实现步骤如下:

(1) 系统参数建立。

给定安全参数 $k \in \mathbb{Z}^+$,由输入的 k 值生成一个素数 q,两个 q 阶群 G_1, G_2,一个可接受的双线性映射 $e: G_1 \times G_1 \rightarrow G_2$。随机选择生成元 $P \in G_1$。取随机数 $s \in \mathbb{Z}_q^*$ 为主密钥,令 $P_{\mathrm{pub}} = sP$。选择两个密码学散列函数 $H_1: \{0,1\}^* \rightarrow G_1^*$ 和 $H_2: \{0,1\}^* \times G_1^* \rightarrow \mathbb{Z}_q^*$。系统参数为 $\mathrm{param} = \langle q, G_1, G_2, e, P, P_{\mathrm{pub}}, H_1, H_2 \rangle$,主密钥为 $s \in \mathbb{Z}_q^*$。

(2) 用户私钥生成。

给定串 $\mathrm{ID} \in \{0,1\}^*$,KGC 计算 $Q_{\mathrm{ID}} = H_1(\mathrm{ID})$,计算私钥 $d_{\mathrm{ID}} = sQ_{\mathrm{ID}}$,这里 s 为主密钥。

(3) 签名生成。

给定私钥 d_{ID} 和消息 m,选取随机数 $r \in \mathbb{Z}_q^*$,计算 $U = rQ_{\mathrm{ID}}$,$h = H_2(m, U)$,$V = (r+h)d_{\mathrm{ID}}$,输出签名 (U, V)。

(4) 签名验证。

对于给定的身份 ID,消息 m 的签名为 (U, V),验证 $e(P, V) = e(P_{\mathrm{pub}}, U + hQ_{\mathrm{ID}})$ 是否成立,这里 $h = H_2(m, U)$。

容易验证正确性:

$$
\begin{aligned}
e(P_{\mathrm{pub}}, U + hQ_{\mathrm{ID}}) &= e(sP, U + hQ_{\mathrm{ID}}) = e(P, sU + shQ_{\mathrm{ID}}) = e(P, srQ_{\mathrm{ID}} + shQ_{\mathrm{ID}}) \\
&= e(P, (r+h)d_{\mathrm{ID}}) = e(P, V)
\end{aligned}
$$

容易发现,该签名机制的构造思想以及 Shamir 基于身份的签名和 ElGamal 签名是一样的,即使用"承诺—挑战—应答"机制构造签名,三者的比较见表 14.1。

显然,类似于基于身份的加密一样的问题(KGC 可以解密任何一个用户的密文),签名方案中 KGC 可以伪造任何一个用户的签名。因此,基于身份的数字签名体制不适合在一个公开的系统环境中应用。换句话说,该签名体制较适合在一个封闭的、KGC 对整个系统中的信息具有法律上的所有权的系统中使用。这是一个非常苛刻的应用环境。

表 14.1 三种签名方案的比较

签名方案	私钥（根据 ID 构造私钥）	签名方程	
		承　诺	挑战（使用 Hash，m，承诺）应答（使用私钥）
Shamir 基于身份的签名	$g=\mathrm{ID}^d \pmod N$	$t=r^e \pmod N$	$s=g\cdot r^{h(t\parallel m)} \pmod N$
Cha-Cheon 基于身份的签名	$d_{\mathrm{ID}}=sQ_{\mathrm{ID}}$	$U=rQ_{\mathrm{ID}}$	$h=H_2(m,U)$，$V=(r+h)d_{\mathrm{ID}}$
ElGamal 签名	$y=g^x \bmod p$	$r=g^k \bmod p$	$s=(h(m)-xr)k^{-1} \bmod (p-1)$

14.4 基于身份的身份识别协议

首先给出基于身份的身份识别(Identity-Based Identification，IBI)协议的一般概念。

基于身份的身份识别协议 IBI＝(Setup，Extract，P，V)由 4 个概率多项式时间的算法组成：系统初始化算法 Setup，私钥提取算法 Extract，两个交互式的算法——证明算法 P 和验证算法 V。

(1)系统初始化算法(Setup)，密钥生成器 KGC 根据系统的安全参数 k 生成系统参数 param 和主密钥 mk，将系统参数 param 公开，主密钥由 KGC 秘密保存。

(2) 私钥提取算法(Extract)，给定一个用户的公开身份 ID，KGC 利用 param、mk 计算该用户的私钥 d，将 d 安全地发送给该用户。

(3) 识别协议：交互式算法 P 和 V，其中证明算法 P 的输入为(param，ID，d)，验证算法 V 的输入为(param，ID)，经过交互后，V 的输出为 T 或 F。

14.4.1 Guillou-Quisquater 的基于身份的身份识别协议

身份识别协议中需要用到权威机构将公钥和身份进行绑定，即提供公钥证书，如果采用身份作为公钥，则不需要公钥证书。下面介绍一个基于身份的身份识别协议——Guillou-Quisquater 的基于身份的身份识别协议。该协议是从原来的 Guillou-Quisquater 协议转化而来的。

当前的协议中直接使用 ID_A 作为公钥，将 TA 的证书发送给 B 改为将 ID_A 发送给 B。验证方程使用公钥 ID_A，而不是原来的公钥 v。

Guillou-Quisquater 的基于身份的身份识别协议实现步骤如下：

(1) 参数设置。

由所有方都信任的权威机构 TA 将身份和公钥绑定，并随机选择类似 RSA 的素数 p 和 q，产生一个模数 $n=pq$。TA 定义一个公钥 $b\geqslant 3$，满足 $\gcd(b,\phi(n))=1$，其中 $\phi(n)=(p-1)(q-1)$，并计算它的私钥 $a=b^{-1} \bmod \phi(n)$。使所有用户都获得系统参数(b,n)。

(2) 私钥提取。

每个实体 A 给定唯一的身份 $\mathrm{ID}_A(1<\mathrm{ID}_A<n)$，$\mathrm{ID}_A$ 中包含了用户 A 的身份信息，如



名字、电话号码等。TA 给 A 秘密分发 $u=((\mathrm{ID_A})^{-1})^a \bmod n$。

(3) 协议消息。

t 轮协议中的每轮有如下 3 条消息(通常 $t=1$)。

$$A \rightarrow B: \mathrm{ID_A}, x = r^b \bmod n$$

$$A \leftarrow B: e(1 \leqslant e \leqslant b)$$

$$A \rightarrow B: y = ru^e \bmod n$$

(4) 协议执行过程如下：

① A 选择随机秘密 r(承诺),$1 \leqslant r \leqslant n-1$,计算(证据)$x=r^b \bmod n$。

② A 发送给 B 整数对($\mathrm{ID_A}, x$)。

③ B 选择一个随机整数 e(挑战),$1 \leqslant e \leqslant b$,发送给 A。

④ A 计算响应 $y=ru^e \bmod n$,发送给 B。

⑤ B 接收 y,计算并判断 $x \equiv \mathrm{ID_A^e} y^b \bmod n$ 是否成立。

完备性分析：如果确实知道秘密信息 s_A,则可以对任意挑战计算响应。易知

$$\mathrm{ID_A^e} \cdot y^b \bmod n = \mathrm{ID_A^e}(ru^e)^b \bmod n = (\mathrm{ID_A} u^b)^e r^b \bmod n = r^b \bmod n = x$$

合理性分析：如果伪装成 A 的攻击者能猜出 B 的挑战 e,则在提交 x 时先选定 y,并取 $x=\mathrm{ID_A^e} \cdot y^b \bmod n$,其后在响应消息中发送选定的 y,则会满足验证方程。这种假冒成功的概率为猜中 e 的概率,即 $1/b$。

表 14.2 给出了两种协议的比较,便于体会两者的区别。

表 14.2 基于证书的和基于身份的 Guillou-Quisquater 身份识别协议的对比

协 议	参 数 设 置	协 议 消 息	协 议 验 证
基于身份	TA 公钥：b TA 私钥：$a=b^{-1} \bmod \phi$ 证明者私钥：$u=((\mathrm{ID_A})^{-1})^a \bmod n$	$A \rightarrow B$: $\mathrm{ID_A}, x=r^b \bmod n$ $A \leftarrow B$: $e(1 \leqslant e \leqslant b)$ $A \rightarrow B$: $y=ru^e \bmod n$	$x \equiv \mathrm{ID_A^e} y^b \bmod n$
基于证书	TA 公钥：b 证明者私钥：u 证明者公钥：$v=(u^{-1})^b \bmod n$	$A \rightarrow B$: $\mathrm{Cert_A}, x=r^b \bmod n$ $A \leftarrow B$: $e(1 \leqslant e \leqslant b-1)$ $A \rightarrow B$: $y=ru^e \bmod n$	$x \equiv v^e y^b \bmod n$

从表 14.2 可知身份验证成立的原因是：$\mathrm{ID_A} u^b = 1 \bmod n$,$vu^b=1 \bmod n$。

14.4.2 Cha-Cheon 基于身份的身份识别协议

Cha-Cheon 的基于身份的身份识别协议与 Cha-Cheon 基于身份的签名十分类似。实现步骤如下：

(1) 参数设置。

令 q 是一个大素数,G 为 q 阶群,其上的 DDH 是困难的,设 P 是群 G 的生成元,选取随机数 $s \in \mathbb{Z}_q$,计算 $P_{pub}=sP$。两个散列函数 $H_1:\{0,1\}^* \times G \rightarrow \mathbb{Z}_q$,$H_2:\{0,1\}^* \rightarrow G$,$e$ 为 G 上的双线性配对。系统参数为 param$=\{G,P,P_{pub},H_1,H_2,e\}$,主密钥为 s。

(2) 私钥提取算法。

给定一个用户的身份 ID,计算公钥 $Q_{ID}=H_2(\mathrm{ID})$,私钥为 $d_{ID}=sQ_{ID}$。

(3) 协议消息。

t 轮协议中的每轮有如下 3 条消息(通常 $t=1$)。

$$P \rightarrow V: \mathrm{ID_P}, U = rQ_{\mathrm{ID}}$$
$$V \leftarrow P: c \in \mathbb{Z}_q$$
$$P \rightarrow V: V = (r+c)d_{\mathrm{ID}}$$

(4) 交互协议如下:

① P 任选 $r \in \mathbb{Z}_q$,计算 $U = rQ_{\mathrm{ID}}$,将 U 发送给 V。

② V 任选 $c \in \mathbb{Z}_q$,发送给 P。

③ P 计算 $V=(r+c)d_{\mathrm{ID}}$,发送给 V。

④ V 验证 $e(P,V)=e(P_{\mathrm{pub}}, U+cQ_{\mathrm{ID}})$是否成立。

思考 14.3 比较 Cha-Cheon 签名方案和 Cha-Cheon 基于身份的身份识别协议。

表 14.3 给出 Cha-Cheon 签名方案和 Cha-Cheon 基于身份的身份识别协议之间的比较。

表 14.3 Cha-Cheon 签名和 Cha-Cheon 基于身份的身份识别协议的对比

方　案	Cha-Cheon 签名	Cha-Cheon 基于身份的身份识别协议
系统参数	$P_{\mathrm{pub}}=sP$ $H_1:\{0,1\}^* \rightarrow G_1^*$ $H_2:\{0,1\}^* \times G_1^* \rightarrow \mathbb{Z}_q^*$ param=〈$q, G_1, G_2, e, P, P_{\mathrm{pub}}, H_1, H_2$〉 $s \in \mathbb{Z}_q^*$	$P_{\mathrm{pub}}=sP$ $H_1:\{0,1\}^* \times G \rightarrow \mathbb{Z}_q$ $H_2:\{0,1\}^* \rightarrow G$ param=$\{G,P,P_{\mathrm{pub}},H_1,H_2,e\}$ $s \in \mathbb{Z}_q$
用户密钥	公钥 $Q_{\mathrm{ID}}=H_1(\mathrm{ID})$ 私钥 $d_{\mathrm{ID}}=sQ_{\mathrm{ID}}$	公钥 $Q_{\mathrm{ID}}=H_2(\mathrm{ID})$ 私钥 $d_{\mathrm{ID}}=sQ_{\mathrm{ID}}$
签名生成(协议消息)	$U=rQ_{\mathrm{ID}}$ $h=H_2(m,U), V=(r+h)d_{\mathrm{ID}}$ 输出签名(U,V)	$P \rightarrow V$: $\mathrm{ID}_P, U=rQ_{\mathrm{ID}}$ $V \leftarrow P$: $c \in \mathbb{Z}_q$ $P \rightarrow V$: $V=(r+c)d_{\mathrm{ID}}$
签名验证	$e(P,V)=e(P_{\mathrm{pub}}, U+hQ_{\mathrm{ID}})$	$e(P,V)=e(P_{\mathrm{pub}}, U+cQ_{\mathrm{ID}})$

容易看到,签名中的挑战是使用散列函数生成的 $h=H_2(m,U)$,而协议中的挑战 c 是验证者选取的,这便是交互零知识证明转化为非交互零知识证明的 FS 启发式方法。

思考 14.4 基于身份的密码系统与基于 PKI 的密码系统的比较。

最后简单总结基于身份的密码系统与基于 PKI 的密码系统的比较。两者都是非对称密码学,主要的不同在于密钥的管理。基于身份的密码学最开始时为了避免使用数字证书而提出的。

(1) 基于身份的密码系统天生具有密钥托管(key escrow)问题,而基于 PKI 的密码系统可以不存在该问题。

(2) 基于身份的密码系统存在私钥泄露,不容易撤销私钥(key revocation and key rollover)。而基于 PKI 的密码系统中,可以使用 CRL(Certificate Revocation List,证书撤销列表)方便地进行公钥证书的撤销。例如基于身份的密码系统不能向基于 PKI 的密

码系统那样生成新的密钥对,因为不能总是改变用户的身份,如果在 ID 上加上一些日期等信息,日期的跨度又不太容易选择。如果跨度太大,用户更改密钥需要等待太久;如果跨度太小,会导致 KGC 工作负担过重。

(3) 基于身份的密码系统在密钥分发(key distribution)时比基于 PKI 的密码系统要简单。公钥从用户身份产生,获得一个人的公钥进行加密和签名变得简单。基于 PKI 的密码系统则需要查看对方的公钥证书,验证 CA 的签名并且检查证书的有效期。

(4) 主密钥安全性。基于 ID 的密码系统中,攻击者如果得到 KGC 的主密钥,就能生成所有私钥,于是可以阅读所有消息和伪造任何签名。在基于 PKI 的密码系统中,CA 的私钥泄露后,攻击者就以 CA 的名义伪造公钥的证书,然后欺骗其他用户。然后,攻击者不能恢复此前生成的私钥,所以只能影响系统未来的安全。

(5) 基于身份的密码系统在注册时需要 KGC 传递私钥给用户,因而需要一个保证机密性和认证性的安全信道。

第 15 章

可证明安全签名和协议*

15.1 可证明安全概述

15.1.1 可证明安全的概念

早期的方案设计在长时间没有找到安全缺陷的基础上被认为是安全的，因此往往陷入发现缺陷、修补，再发现缺陷、修补的恶性循环。为改变这种状况，近年来，提出了可证明安全的思想，可证明安全性成为密码体制安全性的重要理论标准。

可证明安全理论包括可证明安全公钥加密、可证明安全签名和可证明安全协议。该思想也可应用到可证明安全的伪随机密钥发生器、可证明安全的对称密钥加密模式和可证明安全的消息鉴别码。

可证明安全的研究主要依靠两种方法：一种是基于计算复杂性的方法，另一种是基于形式化的方法。后者主要用于可证明安全协议的研究。基于计算复杂性的方法是可证明安全公钥加密和签名的基本方法。依据计算复杂性理论中的“归约”思想，可证明安全试图建立密码学方案与安全假设或困难问题之间的联系，从而将方案的安全性建立在一个公认的“前提基础”之上。

具体而言，可证明安全是指方案的安全性可以被“证明”，这种“证明”用到了反证法和归约的思想。即若可以攻破该方案的安全性，则导致攻破某个被证明是安全的或是被广泛认为是计算上困难的安全假设、密码学原语、数论中的困难问题或一个 NPC 问题的推翻；但是，通常公认为这些安全假设、原语是成立的或问题是困难的，因此原设错误，即不可攻破。从这一意义上讲，可“证明安全”其实就是可“归约”到某个困难问题的假设，因此称为可“归约安全”可能更准确。

可证明安全的意义在于将安全性建立在一个公认的假设基础之上，至少可以排除一些设计上的缺陷，将安全性集中到安全假设之上，这比起没有基于经验和启发式的安全感觉要严格得多，同时，也为密码体制的设计提供了一定的启发式思路。

可证明安全性包含 3 个元素，即安全性定义，安全性所基于的密码学原语或假设，归约证明。安全性定义要首先明确安全的目标，抵抗何种攻击；其次需要明确安全性所给予的密码学原语和假设；最后，通过归约，将一个反命题归约到一个明显的矛盾(如某个数论困难问题的解决等)。

目前的可证明安全加密和签名方案可以分为两大类：一类是在随机预言(random oracle)模型中能证明其安全性，另一类是在标准模型中能证明其安全性。在随机预言模

型中，将密码学散列函数视为是理想的，即当用定义域内某个数问询该散列函数时，从值域内随机选取一个值作为应答；若使用相同的数问询，则使用值域中相同值进行应答。使用不同值进行问询时，应答应是完全独立的。使用随机预言模型证明是安全的方案，当随机预言模型被实际中的散列函数替换后，方案不一定是安全的。但通过这个模型证明是安全的方案比那些尚未证明的方案更让人放心，起码说明方案的其他部分没有安全隐患。实践中的大多数可证明方案都只能在随机语言模型中进行证明，且在这种模型中被证明的方案的效率往往比使用标准模型的方案要高。标准模型是指不假设散列函数是理想的，而采用一些标准的数论假设，如假设因子分解时困难的，或离散对数的计算是困难的。因而标准模型中的安全性证明更加令人信服。但目前的研究表明，在标准模型中的可证明安全方案一般比基于随机预言模型的方案要效率低。

15.1.2　可证明安全的基本思路

通常一个密码学方案的设计有如下几步：

(1) 从现实的安全需求中抽象出需要解决的密码问题的模型。

(2) 深入分析攻击者的能力和可能的攻击目标，应用逻辑合理的定义模式定义安全性。

(3) 提出新的密码学原语，或基于现有的密码学原语来解决问题模型的通用构造方法。

(4) 采用具体的数学结构或安全性假设给出一个通用方案的实例。

(5) 通过可证明安全的方法进行安全证明。

通常，一个安全性证明需要的步骤如下：

(1) 描述一个密码系统以及其操作模式。

(2) 形式化定义一个要达到的安全定义。

(3) 做精确的安全假设。

(4) 给出一个攻破安全定义的算法与攻破安全假设之间的归约证明。这种归约简言之是指，当以攻破安全定义的算法为子函数时，可以构造一个算法攻破安全假设。

定义 15.1　归约。给定两个问题 G 和 H。从 G 到 H 的归约是一个 (t',ε') 算法 SolveG，具有如下性质：

(1) 假定存在 (t,ε) 算法 SolveH 求解问题 H。

(2) SolveG 可以调用 SolveH 并使用它的任一输出值，但 SolveG 不能对于 SolveH 执行的实际运算作任何限定。

(3) 除了访问 SolveH 以外，SolveG 是一个 PPT 算法。

(4) SolveG 可以在时间 t' 内，以 ε' 的概率正确求解问题 G。

几点说明：

(1) $t'=t+f(q,k)$，$\varepsilon'=g(\varepsilon,t,q,k)$，其中，$q$ 是问询次数，k 是安全参数，f，g 是两个分别与 q，k 和 ε，t，q，k 有关的函数。

(2) 如果 G 是 (t',ε') 困难的，则 H 是 (t,ε) 困难的，即如果在时间 t' 内，不能以超过 ε' 的概率求解问题 G，则在时间 t 内，就不能以超过 ε 的概率求解问题 H。

(3) 在时间 $t'\approx t$ 时，ε'与 ε 越接近，归约越紧，即两问题的难度越接近。

主要的计算假设包括：RSA 是单向的(以及强 RSA 问题是困难的)；离散对数问题是困难的；CDH 和 DDH 是困难的；大整数分解时困难的；寻找最短格向量是困难的；判定二次剩余是困难的。

规约证明方法简单地说就是：如果要证明方案的安全性，采用反证法和规约机制，即通过对方案安全性的攻破导致(规约到)某个困难假设的不成立。图 15.1 给出了这种规约证明法的示意图。

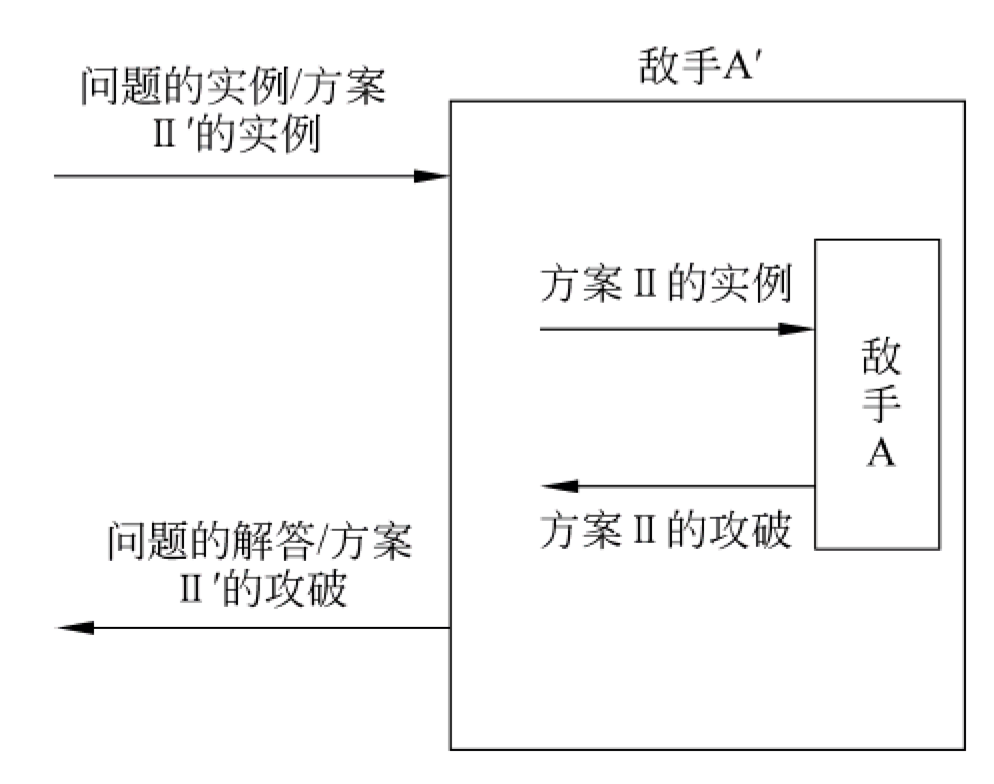


图 15.1 规约证明的一般原理

如果想证明方案Ⅱ的安全性，采取的办法是将攻破方案Ⅱ的敌手视为一个子过程，输入为方案Ⅱ的实例，输出为对该方案的攻破。这种攻破可能表示不同的结果，如 IND-CCA2 中，表示可以输出一个正确的猜测比特，在 EUF-CMA 中表示可以输出一个正确的消息签名对。利用这一子过程构造一个新的过程 A′，该过程的输入是问题的实例，如 RSA 问题、Rabin 问题、离散对数问题等，该过程的输出是对该问题的解答。显然，这一解答是利用子过程 A 的结果。

如果想证明方案Ⅱ′的安全性，采用类似的方法，输入为方案Ⅱ′的实例，调用子过程 A，将方案Ⅱ的实例输入，返回对方案Ⅱ的攻破，利用攻破的结果得到对方案Ⅱ′的攻破。

通常为了规约的方便，可以先固定某些原语(组成部件)是理想的，这样可以排除这些原语外的构成部分是不安全的这一情况，从而将安全的焦点集中到这些原语本身，一个典型的例子是随机预言模型。在随机预言模型下可证明安全，就表明除了该理想模型(理想假设)外，其余部分没有设计上的安全漏洞。

15.2 可证明安全数字签名

15.2.1 数字签名方案的安全性

1.4 节简单介绍过相关概念，这里再次总结一下。敌手的拥有的能力从低到高可以划分成以下几种：

(1) 唯密钥攻击(Key-only Attack，KOA)。敌手拥有公钥以及签名验证函数 $Vrfy_{pk}()$，这是公钥系统提供给敌手的不可避免的能力。

(2) 已知消息攻击(Known Message Attack，KMA)。敌手能够拥有一些消息和签名对。

(3) 选择消息攻击(Chosen Message Attack，CMA)。敌手可以请求一系列消息的签名，他选择消息 $x_1,x_2,\cdots$，并能够得到这些消息的签名。选择只能在挑战消息发出之前进行。

有的文献将选择消息攻击进一步划分为一般选择消息攻击和定向选择消息攻击

(Directed CMA)。区别是前者选择消息是在看到 Alice 的公钥之前,独立于 Alice 的公钥;后者是在看到 Alice 的公钥之后选择消息。还有文献将选择消息攻击中的只能访问一次签名预言机的情况称为单次选择消息攻击(Single Occurrence Chosen-Message Attacks,SO-CMA)。

(4) 自适应选择消息攻击(Adaptive Chosen Message Attack)。敌手可以选择 x_i,而且允许根据先前的访问结果进行后续的选择(选择可以在挑战消息发出之后继续)。

本书中选择消息攻击均指自适应选择消息攻击。

敌手的目标从难到易分为以下几种(安全目标和敌手目标通常相反):

(1) 完全攻破(total break)。敌手能够确定 Alice 的私钥,即签名函数 $\mathrm{Sign}_{sk}()$,从而对任何消息产生有效的签名。从安全的角度而言叫不可攻破性(Unbreakability,UB)。这是签名方案的基本要求。

(2) 完全伪造(universal forgeability)。完全伪造是指敌手在不必拥有私钥的情况下,可以生成任何消息的签名。从安全的角度而言,抗完全伪造则称为完全不可伪造性(Universal UnForgeability,UUF)。

(3) 选择性伪造(selective forgeability)。敌手以某一个不可忽略的概率对另外某个人选择的消息产生一个有效的签名。也就是说,如果给敌手一个消息 x,他能(以某种概率,该概率不可忽略)决定签名,使得 $\mathrm{Vrfy}_{pk}(m,s)=\mathrm{True}$,且该消息 x 是不曾签名过的消息。从安全性角度而言,则称之为选择性不可伪造(Selective UnForgeability,SUF)。

(4) 存在性伪造(existential forgeability)。敌手至少能够为一则消息产生一个有效的签名。换句话说,敌手能产生一个对(m,s),其中 x 是消息,而 $\mathrm{Vrfy}_{pk}(m,s)=\mathrm{True}$。该消息 x 是不曾签名过的消息。即敌手能够伪造一个消息签名对。从安全性角度而言,称为存在性不可伪造(Existential UnForgeability,EUF)。

一个签名方案不可能是无条件安全的,因为对一个给定的消息 x,敌手可以使用公开算法 Vrfy_{pk}测试所有可能的签名 $y\in\phi$,直到发现一个有效的签名。因此,给定足够的时间,敌手总能对任何消息伪造(一个)签名。因此,和公钥加密一样,设计的目标是找到计算上安全的签名方案。

图 15.2 给出了签名方案的安全目标和敌手能力的比较。

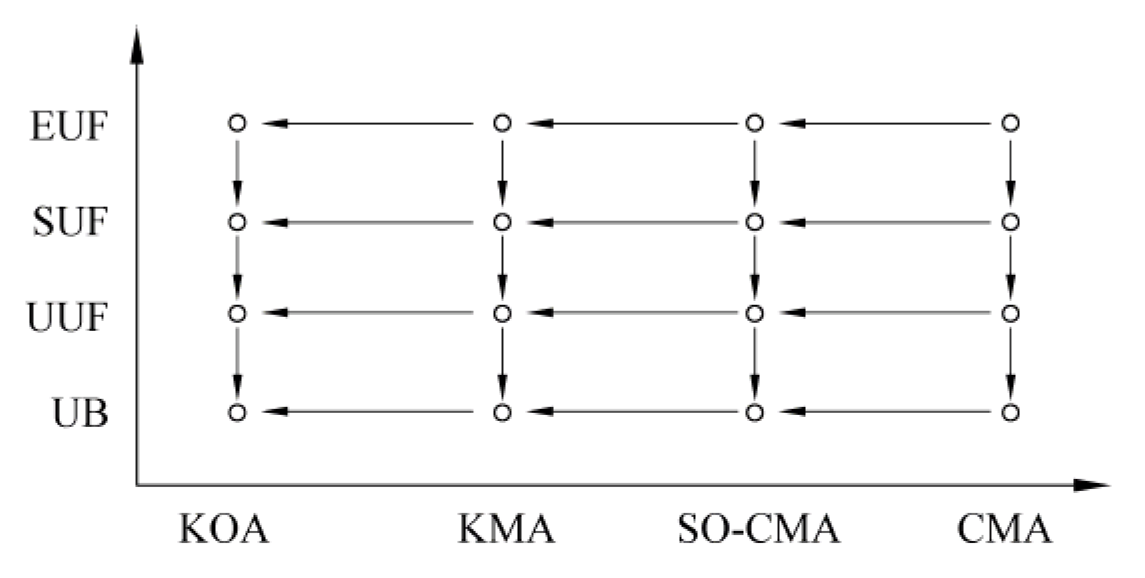


图 15.2 签名方案的安全目标和敌手能力的比较

容易知道,签名方案需要达到的最高安全性是 EUF-CMA。

15.2.2 EUF-CMA安全性的定义

1. 可忽略函数

信息理论安全(information theoretic security)对敌手的计算能力没有假设,因而是理想的。但完善保密的局限是密钥空间必须比明文空间大,且“一次一密”在实际中非常不便。于是想到能否将安全的要求放宽到“实践上不可破译”,从计算上破解是不可行的,即达到计算安全的级别。

基于计算的方法在两个方面“放宽”了完美安全概念:

(1) 安全性仅仅对抗“有效的”敌手,“有效的”是指在可行的时间里运行。

(2) 敌手潜在的成功概率是非常小的(小到可以忽略不计)。

于是,基于计算复杂性理论的现代密码学在讨论任何安全定义时将使用如下的通用形式:如果特定的敌手不能完成特定的攻破,则对给定任务的一个密码学方案是安全的。也就是,令 t,ε 为正数常量,$\varepsilon\leqslant 1$,则一个对安全的具体定义,大体地说,将会是以下的形式:一个方案为(t,ε)安全,如果每个运行时间最多为 t 的敌手以最多为 ε 的概率成功攻破该方案。例如,在最多运行 2^{80} 个 CPU 周期的情况下,没有敌手能够以高于于 2^{-64} 的概率攻破该方案。

与定量方法相对应,还可以采取渐进方法,该方法能够表现出安全性和安全参数(如密钥长度)之间的相对关系。

渐进方法来源于计算复杂性理论,将敌手的运行时间以及成功的概率视为某个参数的函数,而不是具体的数值。通常一个密码方案将包含一个安全参数(如整数 n),当诚实双方初始化方案(即生成密钥),它们选择某个 n 值作为安全参数;假设这个值被任何攻击该方案的敌手知道。那么敌手的运行时间(以及诚实方的运行时间)以及敌手的成功概率将都被看作 n 的函数。于是:

(1) 将“可行的策略”或者“有效的算法”的概念和在以 n 为参数的多项式时间内运行的概率算法等同看待(使用 PPT 来代表概率多项式时间 Probabilisitc Polynormial-Time)。这意味着,对于一些常量 a,c,安全参数为 n,该算法运行时间为 $a\cdot n^c$。要求诚实方在多项式时间内运行,并且仅关心对抗多项式时间的敌手。但是注意,敌手尽管要求在多项式时间内运行,但可能比诚实方的计算能力强(且运行更长时间)。而且,一个需要超多项式时间的攻击策略不被认为是有现实意义的威胁(所以基本上被忽略)。

(2) 将“小的成功概率”的概念和成功概率小于任何以 n 为参数的多项式的倒数等同看待,这意味着对于每个常量 c,当 n 的值足够大时,敌手成功的概率小于 n^{-c}。比任何多项式的倒数增长得都慢的函数被称为可忽略的(negligible)。

可忽略的成功概率是一个基于计算的安全中的十分关键的概念。现代密码学允许将敌手能以非常小的概率攻破的方案仍视为“安全的”。考虑到多项式运行时间是可行的,类似地,认为概率为多项式的倒数是显著的(足够大的),因此,对某个(正)多项式 p,如果一个敌手能够成功地攻破该方案的概率为 $1/p(n)$,则该方案被认为是不安全的。但是,对任意多项式 p,如果攻破该方案的概率渐进地小于 $1/p(n)$,则认为该方案是安全的。这是因为敌手成功的概率太小,被认为是无趣的。这样的概率叫做可忽略的,其严格的定

义如下：

定义 15.2　可忽略函数的定义。函数 f 为可忽略的，如果对于每个多项式 $p(\cdot)$，存在一个 N，使得对于所有的整数 $n>N$，都满足 $f(n)<\frac{1}{p(n)}$。

一个和上面等价的公式化的表述是，对所有常量 c，存在一个 N，使得对于所有的 $n>N$，满足 $f(n)<n^{-c}$。为了方便记忆，上面的表述也可以陈述如下：对于任意多项式 $p(\cdot)$ 以及所有足够大的 n 值，满足 $f(n)<\frac{1}{p(n)}$。通常将一个可忽略函数表示为 negl。

也可以陈述为，存在一个多项式 $p(\cdot)$，满足 $|f(n)|=\Omega\left(\frac{1}{p(n)}\right)$。

因此，对渐进安全的定义一般采用下面的形式：一个方案是安全的，如果每个 PPT 敌手以可忽略的概率成功攻破该方案。具体而言，一个方案是安全的，如果每个概率多项式时间的敌手执行某个特定类型的攻击，对于每个多项式 $p(\cdot)$，存在一个整数 N，对 $n>N$，敌手在这次攻击中成功的概率小于 $\frac{1}{p(n)}$。

2. 数字签名的定义

定义 15.3　数字签名方案的定义。一个数字签名方案是一个由概率多项式时间算法组成的三元组(Gen，Sign，Vrfy)，满足下列条件：

(1) 密钥生成算法 Gen 以安全参数 1^n 为输入，并且输出一对密钥(pk，sk)，分别称为公钥和私钥。为了方便，假设 pk 和 sk 长度至少为 n，并且 n 可以由 pk，sk 确定。

(2) 签名算法 Sign 以一个私钥 sk 和消息 $m\in\{0,1\}^*$ 作为输入。输出一个签名 σ，表示为 $\sigma\leftarrow \mathrm{Sign}_{sk}(m)$。

(3) 确定的验证算法 Vrfy 以一个公钥 pk、消息 m 和一个签名 σ 作为输入。输出一位 b，当 $b=1$ 时意味着签名有效，当 $b=0$ 时为签名无效。将其表示为 $b:=\mathrm{Vrfy}_{pk}(m,\sigma)$。

此方案需要对于任意 n，所有由 Gen(1^n)输出的(pk，sk)以及任意 $m\in\{0,1\}^*$ 必须满足

$$\mathrm{Vrfy}_{pk}(m,\mathrm{Sign}_{pk}(m)) = 1$$

如果(Gen，Sign，Vrfy)对于所有由 Gen(1^n)输出的(pk，sk)，算法 Sign_{sk} 只对消息 $m\in\{0,1\}^{l(n)}$ 有定义(并且 Vrfy_{pk} 对于任意 $m\notin\{0,1\}^{l(n)}$ 输出都为 0)，此时称(Gen，Sign，Vrfy)是一个消息长度为 $l(n)$ 的数字签名方案。当 $\mathrm{Vrfy}_{pk}(m,\sigma)=1$ 时，称 σ 是消息 m 的一个有效签名(关于某个公钥 pk)。

3. 数字签名 EUF-CMA 安全的定义

有了可忽略函数和数字签名方案的定义之后，下面给出 EUF-CMA 安全的定义。

给定一个由签名者 S 生成的公钥 pk，如果敌手输出一个消息 m 和有效签名 σ，并且先前 S 并没有给 m 签名，称敌手输出一个“伪造”签名。类似在消息鉴别中，数字签名的安全性意味着，即使敌手允许得到很多自己选择消息的签名，敌手也不能输出一个伪造。这和消息鉴别码安全性的定义形成了类比。

设 $\Pi=$(Gen，Sign，Vrfy)是一个数字签名方案，对于敌手 A 和参数 n，考虑下面的伪造数字签名实验 $\mathrm{Sig}_{A,\Pi}^{\text{euf-cma}}(n)$：

(1) 运行 Gen(1^n)得到密钥(pk,sk)。

(2) 将 pk 和访问"签名预言机"Sign(·)的权利给敌手 A(此预言机对于敌手选定的任意消息 m 返回一个签名 $\mathrm{Sign}_{sk}(m)$)。然后敌手输出(m,σ)。设 Q 表示 A 在执行期间问询签名的消息的集合。

(3) 当且仅当 $\mathrm{Vrfy}_{pk}(m,\sigma)=1$,且 $m\notin Q$ 时实验输出定义为 1。

定义 15.4 签名方案的 EUF-CMA 安全。一个签名方案 Π=(Gen,Sign,Vrfy)如果对任意概率多项式时间敌手 A 存在一个可忽略函数 negl,满足

$$\Pr[\mathrm{Sig}_{A,\Pi}^{\text{euf-cma}}(n)=1]\leqslant \mathrm{negl}(n)$$

则此签名方案在适应性选择消息攻击下是存在性不可伪造(EUF-CMA)。

这里给出示意图(图 15.3)来方便理解。通常使用攻击者 A 和系统(也成为挑战者)之间的游戏来描述 EUF-CMA 安全性。对签名体制(Gen,Sign,Vrfy),定义安全的交互游戏(game)如下:

(1) 系统运行 Gen(1^n)产生密钥对(pk,sk),将 pk 给 A。

(2) A 选择合理规模的一系列明文,系统给出对应的签名。

(3) A 输出未曾问询的消息 m^* 的签名 s^*。

定义敌手 A 的伪造优势为 $\mathrm{Adv_A}=|\Pr[\mathrm{Vrfy}(pk,m^*,s^*)=1]-1/2|$。如果对一个签名体制而言敌手的伪造优势(用安全参数表示)是可忽略的,则称该签名体制是 EUF-CMA 安全的。

用集成的符号可表示如下:

$$\mathrm{Adv}_A^{\text{euf-cma}}=|\Pr_{r\xleftarrow{R}U}[(sk,pk)\leftarrow \mathrm{Gen}(1^n),$$

$$S=\{s_1,\cdots,s_j\}\leftarrow\{\mathrm{Sign}(sk,r,m_1),\cdots,\mathrm{Sign}(sk,r,m_j)\},$$

$$(m^*,s^*)\leftarrow A(s_1,\cdots,s_j,pk),m^*\notin S:\mathrm{Vrfy}(pk,m^*,s^*)=1]-1/2|<\varepsilon$$

为了突出签名的随机性,这里在签名生成算法中使用了随机数 r。去掉绝对值符号,可变为如下形式:

$$\mathrm{Adv}_A^{\text{euf-cma}}=2\times\Pr_{r\xleftarrow{R}U}[(sk,pk)\leftarrow \mathrm{Gen}(1^n),$$

$$S=\{s_1,\cdots,s_j\}\leftarrow\{\mathrm{Sign}(sk,r,m_1),\cdots,\mathrm{Sign}(sk,r,m_j)\},$$

$$(m^*,s^*)\leftarrow A(s_1,\cdots,s_j,pk),m^*\notin S:\mathrm{Vrfy}(pk,m^*,s^*)=1]-1<\varepsilon$$

15.2.3 随机预言模型

随机预言模型是散列函数的一种理想化模型,该模型对于每一个消息都试图得到一个理想的散列函数值。在这个模型中,随机从 $F^{X,Y}$ 中选出一个散列函数 $H:X\to Y$,仅仅允许预言器访问函数 H,这表示对于任一消息 x,随机预言模型都不会给出一个公式或者算法来计算消息摘要 $H(x)$的值。因此,计算 $H(x)$ 值的唯一方法是问询预言机,这种散列函数模型相当于根据给出的消息 x,在随机数表中查询 $H(x)$的值,对于每一个 x,都有随

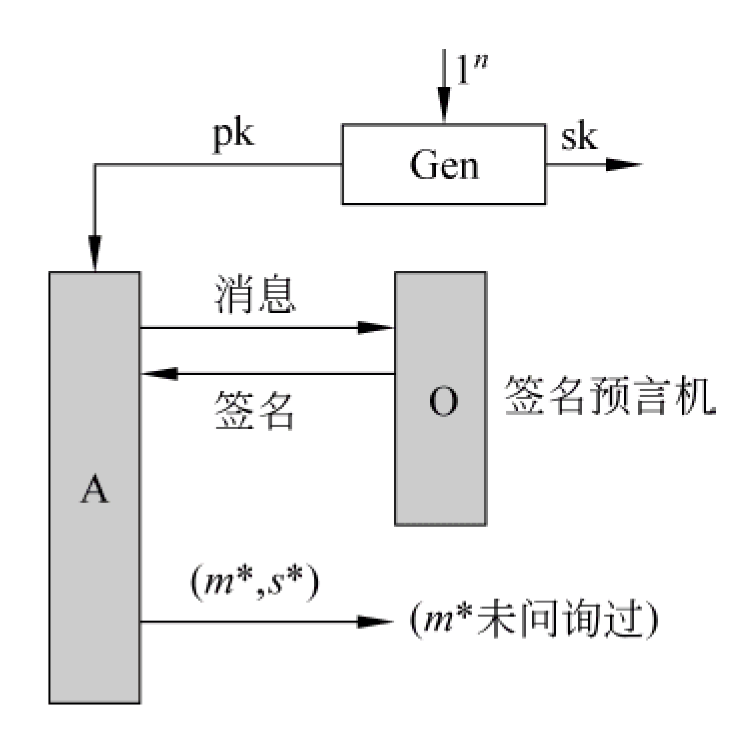


图 15.3 EUF-CMA 安全性

机的 $H(x)$ 与之对应。

定义 15.5 随机预言模型的定义。令 $F^{X,Y}$ 是所有从 X 到 Y 的函数集合。假定 $|X|=N,|Y|=M$，则 $|F^{X,Y}|=M^N$。任何散列函数簇 $F\subseteq F^{X,Y}$ 被称为一个 (N,M) 散列簇。

对于随机预言模型，以下结论成立。

定理 15.1 随机选择的 $H\in F^{X,Y}$，并取子集 $X_0\subseteq X$，假设当且仅当 $x\in X_0$ 时，$H(x)$ 的值 t 由问询预言机确定，那么对于所有的 $x\in X_0$ 和 $y\in Y$，$\Pr(H(x)=y)=1/M$。

以上结论中，概率 $\Pr(H(x)=y)$ 表示对任意的 $x\in X$，计算函数 $H(x)$ 的值等于指定值的条件概率。通过该结论可知，M 的值越大，产生碰撞的概率就越小。

随机预言机的确定性和均匀输出特性意味着随机预言机输出的熵大于其输入的熵，但是，根据香农的熵理论，一个确定函数绝不可能"放大"熵。因此，现实中不存在随机预言机。

15.2.4 RSA-FDH

FDH 是指全域散列函数(Full Domain Hash，FDH)，即其值域与陷门单向置换的定义域相同，安全性证明中需要 FDH 满足随机预言模型。

令 GenRSA 如前所述，令 H 为函数，其定义域为 $\{0,1\}^*$，值域为 $\mathbb{Z}_N^*$(对任意的 N)。如下构造一个 RSA-FDH 签名方案：

(1) Gen：输入 1^n，运行 GenRSA(1^n) 来计算 $\langle N,e,d\rangle$，令 H 的值域为 $\mathbb{Z}_N^*$。公钥为 $\langle N,e\rangle$，私钥为 $\langle N,d\rangle$。

(2) Sign：输入一个私钥 $\langle N,d\rangle$ 以及一个消息 $m\in\{0,1\}^*$，计算

$$\sigma:=[H(m)^d \bmod N]$$

(3) Vrfy：输入一个公钥 $\langle N,e\rangle$、一个消息 m 以及一个签名 σ，当且仅当 $\sigma^e \overset{?}{=} H(m) \bmod N$ 时，输出 1。

定理 15.2 如果假设 RSA 问题是困难的，H 为随机预言机，那么该构造方案是适应性选择消息攻击下存在性不可伪造的(EUF-CMA 安全)。

证明：令 $\Pi=(\text{Gen},\text{Sign},\text{Vrfy})$ 表示构造方案，令 A 为一个概率多项式时间敌手。定义

$$\varepsilon(n) \overset{\text{def}}{=} \Pr[\text{Sig-forge}_{A,\Pi}(n)=1]$$

为了方便，描述实验 $\text{Sig-forge}_{A,\Pi}(n)$ 的步骤：

(1) 选择一个随机函数 H。

(2) 运行 GenRSA(1^n) 来获得公钥和私钥 (N,e,d)。

(3) 敌手 A 被指定 pk $=\langle N,e\rangle$，可以问询 $H(\cdot)$ 以及签名预言机 $\text{Sign}_{\langle N,d\rangle}(\cdot)$。当 A 请求一个消息 m 的签名时，它被指定回复 $\sigma:=[H(m)^d \bmod N]$。

(4) A 输出一对值 (m,σ)，其中 A 之前没有请求 m 的签名。如果 $\sigma^e=H(m) \bmod N$，该实验的输出为 1，否则为 0。

为了简化，不失一般性地假设：①A 未向随机预言机提出两次相同的问询；②如果 A 请求消息 m 的签名，那么它在之前已经问询过 $H(m)$；③如果 A 输出 (m,σ)，那么它之

前已问询过 $H(m)$。令 $q=q(n)$ 为 A 问询随机预言机的次数的上界。考虑下面的算法 A′。

算法 A′:

该算法的输入指定为(N,e,y^*)。

(1) 选择 $j\leftarrow\{1,\cdots,q\}$。

(2) 指定 A 公钥 $pk=\langle N,e\rangle$。在表格中存储三元组(·,·,·),初始值为空。一个条目(m_i,σ_i,y_i)指示出:A′已经令 $H(m_i)=y_i$ 以及 $\sigma_i^e=y_i \bmod N$。

(3) 当 A 向其随机预言机问询 $H(m_i)$时,如下作答:

① 如果 $i=j$,返回 y^*。

② 否则,选择一个随机值 $\sigma_i\leftarrow\mathbb{Z}_N^*$,计算 $y_i:=[\sigma_i^e \bmod N]$,将 y_i 作为答案返回,并且在表格中存储(m_i,σ_i,y_i)。

当 A 请求消息 m 的签名时,令 i 满足 $m=m_i$,如下回答问询:①

① 如果 $i\neq j$,那么在表格中,存在一个条目 m_i,σ_i,y_i。返回 σ_i。

②如果 $i=j$,那么中止该实验。

(4) 在 A 执行的最后,输出(m,σ)。如果 $m=m_j,\sigma^e=y^* \bmod N$,那么输出 σ。

很显然,A′是在多项式时间内运行的。假设 A′的输入通过运行 GenRSA(1^n)来生成,获得(N,e,d),然后均匀随机地选择 $y^*\leftarrow\mathbb{Z}_N^*$。在第一步中,由 A′选择出来的标志 j 代表着一个猜测:A 的哪一个预言机问询将会对应着 A 最后输出的伪造签名。当该猜测是正确的时候,A 作为一个子程序被 A′在实验 RSA-$\text{inv}_{A',\text{GenRSA}}(n)$中运行时,以及 A 在实验 Sig-$\text{forge}_{A,\Pi}(n)$中所见视图的分布是相同的。这是因为:当作为一个子程序被 A′运行时,A 的 q 个随机预言机问询中的每一个都确实是用随机值来回答的:

(1) 问询 $H(m_j)$是用 y^* 来回答的,y^* 是从$\mathbb{Z}_N^*$中均匀随机选择出来的。

(2) 问询 $H(m_i)(i\neq j)$是用 $y_i=[\sigma_i^e \bmod N]$来回答的,其中 σ_i 是从$\mathbb{Z}_N^*$中均匀随机选择的。因为 RSA 是一个置换,这意味着 y_i 在$\mathbb{Z}_N^*$也是均匀分布的。

此外,j 和 A 的分布是相互独立的,除非 A 正好请求了消息 m_j 的签名。但是,在这种情况下,A′的猜测是错误的(因为一旦 A 请求了 m_j 的签名,它就不能输出一个消息 m_j 的伪造签名)。

当 A′猜测正确,且 A 输出了一个伪造签名,那么 A′就解决了 RSA 难题的指定实例(因为 $\sigma^e=y^* \bmod N$,因此 σ 是 y^* 的逆)。因为 A′正确猜中的概率是 $1/q$,有

$$\Pr[\text{RSA-inv}_{A',\text{GenRSA}}(n)=1]=\varepsilon(n)/q(n)$$

因为 RSA(与 GenRSA 相关)是困难的,存在着一个可忽略的函数 negl 满足

$$\Pr[\text{RSA-inv}_{A',\text{GenRSA}}(n)=1]\leqslant \text{negl}(n)$$

因为 q 是多项式,于是 $\varepsilon(n)$也是可忽略的,证毕。

① 这里 m_i 表示的是向随机预言机的提出的第 i 个问询。前面已经假设,如果 A 请求一个消息的签名,那么它之前已经向散列随机预言机询问过该消息了。

15.3　可证明安全协议简介

在研究协议的安全性时，通常假设其中使用的基本密码算法（如加密算法、消息鉴别码和签名算法等）是安全的，并把重点放在协议自身结构的安全上。早期的协议安全性研究通过攻击检测进行，即看协议是否能够抵抗已知的各种攻击。后来产生了形式化分析（如基于各种逻辑推理）的思想和各种理论。

形式化方法是指用数学方法描述和推理基于计算机的系统，在技术上通过精确的数学手段和强大的逻辑分析工具得到支持。Needham 和 Schroeder 首次提出了使用形式化的方法对协议进行分析的思想。1981 年 Dolev 和 Yao 发表了第一篇用形式化方法对协议进行分析的文章。随后出现了一系列用多项式时间算法对特定类型协议的安全性进行分析的方法。1990 年 Burrows、Abadi 和 Needham 提出了一种基于逻辑表示和推理的方法，称为 BAN 逻辑，此后出现了大量的逻辑分析方法。1998 年 Paulsont 提出了归纳证明法。1999 年 Abadi 提出了类型检测方法。2001 年 Song 等人利用串空间原理发展了一种自动检测工具。

近年来，随着可证明安全理论的兴起和在算法设计中的成功应用，协议的可证明安全理论也应运而生，成为协议安全性分析的另外一个重要思路。基于计算复杂性的方法采用公理集合论的方法，将安全建立在一个计算复杂性理论这一基础之上，从密码学意义上讲有着坚实的基础，而大多数建立于 Dolve-Yao 模型之上的形式化方法没有真正建立起这种密码学的可靠性。因此，将这两种方法统一到一个框架之中，建立形式化方法的可靠性成为目前备受关注的研究内容，将形式化分析和基于计算复杂性的安全性证明结合起来的思路可能会在两种方法中取长补短。

这里简要介绍一个可证明安全认证协议的例子。

Bellare 和 Rogaway 在 1993 年和 1995 年分别给出了可证明安全性的两方和三方会话密钥分配协议。协议本身很简单，但其意义在于首次建立了会话密钥分配协议的形式化安全模型，称为 BR 安全模型。这是一种现实模型，即协议只定义在现实世界中，而安全性是通过会话密钥与某随机数的计算不可区分性来定义的（基本方法是使会话密钥和一个敌手易于得到的某伪随机数紧密联系。）

协议定义：$P=(\Pi,\Psi,\mathrm{LL})$，这是一个多项式时间可计算的三元组函数，Π 描述诚实方的行为，Ψ 描述可信中心 S 的行为，LL 描述用户主密钥的初始分布。

协议参与方：$I=\{0,1,2,\cdots,N\}$（0 代表 S），敌手 E。协议参与者的所有预言机只能被动地回答攻击者 E 对它们进行的各种问询，预言机之间并不进行直接的通信。也就是说，模型中至少存在一个良性（benign）攻击者，它唯一的动作就是忠实地传递预言机之间的协议消息。

敌手模型：E 控制所有合法方之间的通信（如可以控制协议启动时间、篡改、替换或删除数据等），形式化为一个概率图灵机，并且它能够访问模型中的所有预言机，即 Oracle $\Pi_{i,j}^{s}$ 和 $\Psi_{i,j}^{s}$，其中 $i,j\in I$。$\Pi_{i,j}^{s}$ 形式化了成员 i 试图和成员 j 协商一个会话密钥的通信实例，$\Psi_{i,j}^{s}$ 形式化了 S 试图给 i、j 分配会话密钥的通信实例。

E 所能发起的攻击形式化为 5 种预言机问询：(SendPlayer,i,j,S,x)；(SendS,i,j,S,x)；(Reveal,i,j,S)；(Corrupt,i,K)；(Test,i,j,S)。前 4 种预言机问询可以是多项式形式的，形式化了 E 所能发动的各种攻击，如窃听、已知会话密钥、重放、收买某合法方等攻击，(Test,i,j,S)则只能问询一次，是为了定义安全性而提供的“测试”预言机。

(SendPlayer,i,j,S,x)：E 可以向预言机 $\Pi_{i,j}^{S}$ 发送消息。该预言机按照协议规范应答一个响应消息。

(SendS,i,j,S,x)：E 可以向预言机 $\Psi_{i,j}^{S}$ 发送消息。该预言机按照协议规范应答一个响应消息。

(Corrupt,i,K)：此问询要求被问询的协议参与者返回它拥有的长期私钥。相应地，回答过 Corrupt 问询的实体的状态被称为“已攻陷”(corrupted)。

(Reveal,i,j,S)：收到此问询的预言机返回它协商得到的会话密钥。如果该预言机的状态还不是“已接受”(accepted)，那么返回一个符号$\perp$表示终止。

(Test,i,j,S)：E 可以向一个预言机发出 Test 问询。E 将收到该预言机所拥有的会话密钥或者一个随机值。具体来说，该预言机通过投掷一枚公平硬币 $b\in_R\{0,1\}$来回答此查询。若投币结果为 0，那么它返回会话密钥的比特长度。

安全性定义：敌手的成功是以其在进行 Test 问询之后区分会话密钥与随机密钥的优势来衡量的。除合理性(validity)之外，敌手得到预言机问询(Test,i,j,S)的回答后，对挑战进行“猜测”，得到正确应答的优势函数为 $\mathrm{Advantage}_{P,F,S_n}^{E}(k)=(2\Pr[\mathrm{GoodGuess}]-1)$，如果该值是可忽略的，则称协议是安全的。这里，当敌手发起 Test 询问时，预言机回答如下：$b\in_R\{0,1\}$，如果 $b=0$，返回一个随机数，否则返回敌手要得到的会话密钥；如果敌手对 b 的取值猜测正确，就称事件 GoodGuess 发生，敌手成功。敌手随机猜测成功的概率为 1/2。上述定义就是表明，如果敌手猜测成功的概率偏离 1/2 的值是可忽略的，则敌手失败(即敌手不具有好于随机猜测的能力，于是不能区分会话密钥与随机密钥)，即意味着协议是安全的。

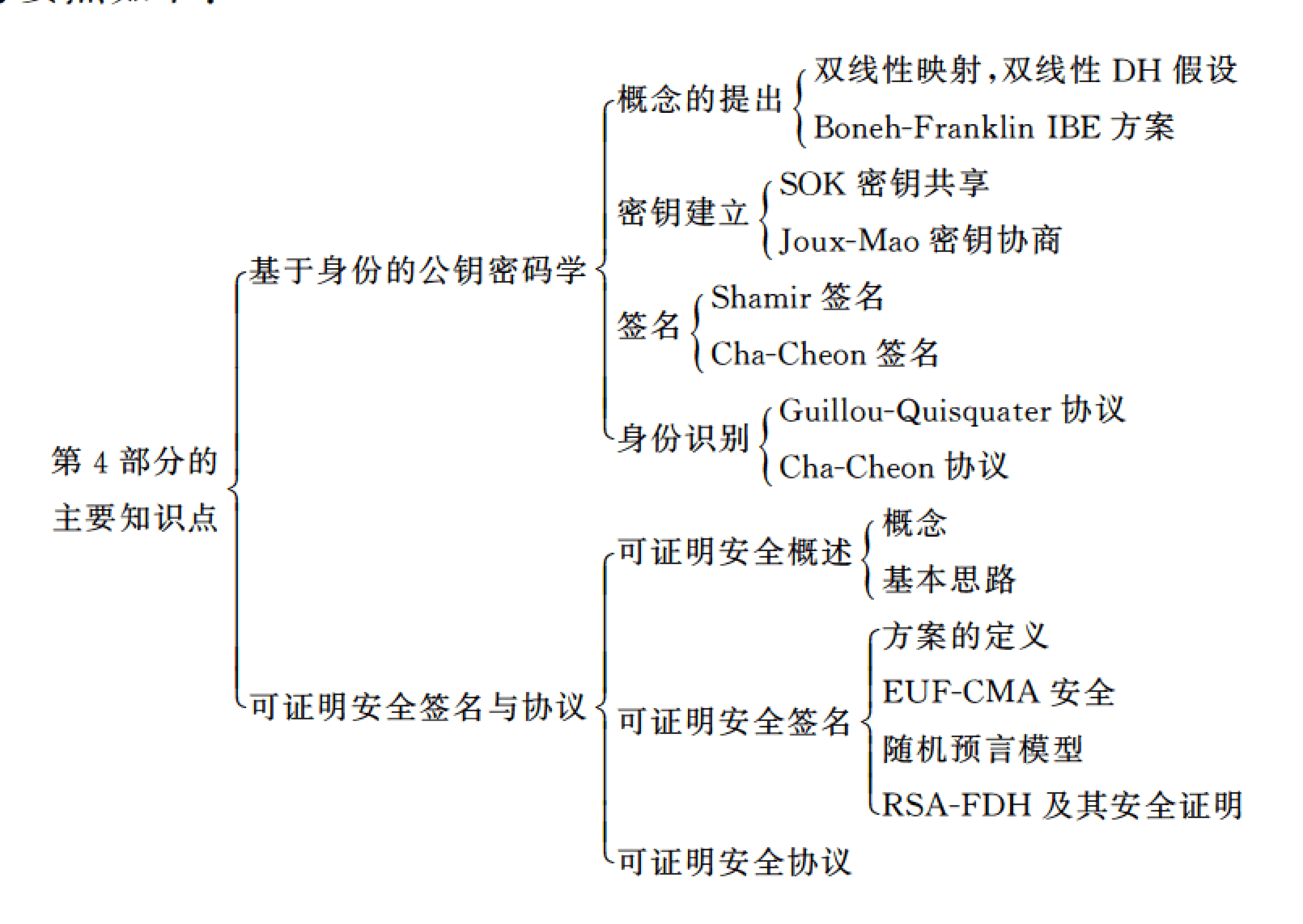
第 4 部分

小　结

本部分介绍了基于身份的公钥密码学，包括加密、密钥共享、密钥协商、签名、身份识别协议。另外还介绍了可证明安全公钥密码学的基础知识以及可证明安全数字签名和协议。

本部分的重点是基于身份的签名、基于身份的密钥共享体制、RSA-FDH，难点是双线性映射假设、可证明安全性、EUF-CMA 安全性。

本部分要点如下：



扩展阅读建议

(1) 关于数字签名的可证明安全性

J. Katz. 数字签名. 任伟,译. 北京:国防工业出版社,2013.

(2) 关于可证明安全性

J. Katz, Y. Lindell. 现代密码学——原理与协议. 任伟,译. 北京:国防工业出版社,2012.

(3) 基于身份的密码学

Adi Shamir. Identity-Based Cryptosystems and Signature Schemes. Advances in Cryptology: Proceedings of CRYPTO 84, Lecture Notes in Computer Science, 1984(7): 47-53.

Dan Boneh, Matt Franklin. Identity-based Encryption from the Weil Pairing, CRYPTO 2001 (Springer) 2139/2001: 213-229.

Marc Joye, Gregory Neven, Eds. Identity-Based Cryptography. IOS Press, 2009.

张方国,陈晓峰. 基于身份的密码体制的研究综述//中国密码学发展报告 2008. 北京:电子工业出版社,2009.

(4) 密码学基础

Blum, Micali. How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits. SIAM Journal on Computing, 1984(13): 850-864 (给出了一个基于硬核谓词的通用的伪随机比特生成方案)

L. Blum, M. Blum, M. Shub. A Simple Unpredictable Pseudo-Random Number Genereator. SIAM Journal of Computing, 1986(15): 364-384 (伪随机比特生成器 BBS 方案)

A. C. Yao. Theroy and Applications of Trapdoor Functions. FOCS82, 80-91(基于陷门置换构造伪随机生成器)

L. A. Levin. One-way Functions and Psuedo-Random Generators. STOC85, 363-365(基于任意单向函数构造伪随机生成器)

Goldreich, S. Goldwasser, S. Micali. How to Constructe Random Functions. FOCS84, 464-479(扩展后发表在 Journal of ACM, 1986(33): 792-807,构造伪随机函数的方法)

Goldreich, S. Goldwasser, S. Micali. On the Cryptogarphic Applications of Random Functions, Crypto84, 1985: 276-288. (伪随机函数在可证明安全协议上的应用)

Luby，Rackoff. How to Construct Pseudo-Random Permutations from Pseudo-Random Functions. SIAM Journal on Computing，1988(17)：373-386.(从伪随机函数构造伪随机置换)

R. Impagliazzo，L. Levin，M. Luby. Pseudo-Random Generation from One-Way Functions. STOC89. 12-24（证明了单向函数的存在是安全伪随机生成器存在的充分必要条件）

(5) 可证明安全

M. Naor，M. Yung. Public-Key Cryptosystems Provably Secure against Chosen Ciphertext Attacks，STOC90，427-437(第一个 IND-CCA1 安全的公钥加密)

C. Rockoff，D. R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. Crypto91(第一个 IND-CCA2 安全的公钥加密)

Damgard. Towards Practicial Public Key Systems Secure Against Chosen Ciphertext Atacks. Crypto91，1992：445-456.

Y. Zheng，J. Sebeerry. Immunizing Public Key Cryptosystems against Chosen Ciphertext Attacks. IEEE JSAC，1993(11)：715-724.

T. Okamoto. Efficient Blind and Partially Blind Signatures without Random Oracles. TCC 2006，2006，LNCS3876，80-99(基于标准模型构造的效率)

J. Coron，J. Patarin，Y. Seurin. The Random Oracle Model and the Ideal Cipher Model Are Equivalent. CRYPTO08，LNCS5157，1-20.（ROM 和 ICM 之间的等价性）

D. Dolev，C. Dwork，M. Naor. Non-Malleable Cryptography. ACM STOC91，1991：542-552(NM 概念的提出)

M. Bellare，A. Desai，D. Pointcheval. Relations among Nations of Security for Public-Key Encryption Schemes. Crypto'98，1998，LNCS1462，26-45.（IND-CCA2 和 NM-CCA2 的等价性）

E. Kiltz. Chosen-Ciphertext Security from Tag-Based Encryption. TCC，LNCS3876，Springer-Verlag，2006：581-560（IND-CPA2 转化为 IND-CCA2）

R. Canetti，S. Halevi，J. Katz. Chosen-Ciphertext Security from Identity-based Encryption. Proceedings of Eurocrypt 2004，Springer-Verlag，LNCS2004，2004：207-222.（IND-CPA2 转化为 IND-CCA2）

M. Bellare，P. Gogaway. Random Oracles are Practical：A Paradigm for Designing Efficient Protocols，CCS93，1993：62-73.（ROM 的提出）

M. Bellare，P. Gogaway. Optimal Asymmetric Encryption，CRYPTO'94，Berlin，Springer-Verlag，1993：92-111.（OAEP 的提出，明文敏感的概念的提出）

E. Fujisaki，T. Okamoto，D. Pointcheval et al. RSA-OAEP Is Secure under the RSA Assumption. Crypto'01，LNCS2139，Springer-Verlag，2001，：260-274.（OAEP 的补救）

M. Bellare，P. Rogaway. The Exact Security of Digital Signatures—How to Sign with RSA and Rabin. Eurocrypt96，LNCS1070，Spring，1996：399-416(利用 ROM 得

到可证明安全 RSA 和 Rabin）

E. Fujisaki，T. Okamoto. How to Enhance the Security of Public-Key Encryption at Minimum Cost. PKC99，Springer，1999：53-68(FO 变换)

D. Pointcheval. Chosen-Ciphertext Security for any One-way Crtyptosystem. PKC00，Springer，2000：129-146(P00 变换)

T. Okamoto，D. Pointcheval. REACT：Rapid Enhanced-Security Asymmetric Cryptosystem Transform，CT-RSA01，Springer，2001：159-175(OP01 变换，REACT 的提出)

R. Cramer，V. Shoup. A Practicial Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. Crypto98，Spinger，1998：13-25(CS98 方案)

V. Shoup. Using Hash Functions as a Hedge against Chosen Ciphertext Attack. Eurocrypt00，Springer，2000：275-288(CS98 方案的改进，S00 方案)

R. Cramer，V. Shoup. Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Seccure Public-Key Encryption. Eurocrypto02，Springer，2002：45-64 (S00 方案的进一步改进，CS02 方案)

M. Abdalla，M. Bellare，P. Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. CT-RSA01，Springer，2001：143-158(DHIES 的提出)

D. Boneh，M. Franklin. Identity Based Encryption from the Weil Pairing. CRYPTO，2001，LNCS2139，2001：213-229.(第一个基于 ROM 的安全 IBE)

K. Kurosawa，Y. Desmedt. A New Paradigm of Hybrid Encryption Scheme. Crypto04，Springer，2004：426-442 (高效率的标准模型下的 IND-CCA2 安全方案)

D. Boneh，X. Boyen. Effcient Selective-ID Secure Identity Based Encryption without Random Oracles. EUROCRYPT 2004，LNCS3027，Springer-Verlag，2004：223-238.(第一个基于标准模型的安全 IBE，Selective-ID 安全，BB 方案)

R. Canetti，S. Halevi，J. Katz. Chosen-Ciphertext Security from Identity-based Encryption. Eurocrypt04，Springer，2004：207-222 (CHK 方案)

D. Boneh，X. Boyen，E. Goh. Hierarchical Identity-based Encryption with Constant Ciphertext. EuroCrypt'05，LNCS3494，Springer-Verlag，2005：440-456.(可证明安全层次 IBE)

B. Waters. Effcient Identity-based Encryption without Random Oracles. Advances in Cryptology-urocrypt'05，LNCS3494，2005：114-127.(Water 提出的加密和签名方案)

T. Okamoto，D. Pointcheval. EPOC-3：Efficient Probabilistic Public-Key Encryption (Version 2). Contribution to IEEE - describes EPOC-3. http://grouper.ieee.org/groups/1363/StudyGroup/NewFam.html#epoc3.(EPOC)

T. Okamoto，S. Uchiyama. A New Public-Key Cryptosystem as Secure as Factoring. Advances in Cryptology - EUROCRYPT '98 Proceedings，1998(1403)：308-

318.

(6) 安全签名

Shafi Goldwasser, Silvio Micali, Ronald Rivest. A Digital Signature Scheme Secure against Adaptive Chosen-Message Attacks, SIAM Journal on Computing, 1988,17(2): 281-308(MR 签名,第一个 EUF-CMA 签名,用无爪置换对构造)

M. Bellare, S. Micali. How to Sign Given Any Trapdoor Function. CRYPTO88, 1988: 200-215 (用陷门单向置换构造签名)

M. Naor, M. Yung. Universal One-Way Hash Functions and their Cryptographic Applications. STOC89, 1989: 33-43(通用散列函数的提出, UOWHF 的存在当且仅当单向置换存在,UOWHF 可构造可证明安全的签名)

J. Rompel. One-Way Functions are Necessary and Sufficient for Secure Signatures. STOC90, 1990: 387-394 (安全签名的充分必要条件是单向函数的存在性)

D Pointcheval, J Stern. Security Proofs for Signature Schemes. Eurocrypt '96. Springer-Verlag, 1996: 387-398(Pointcheval-Stern 签名,提出了 Forking 引理)

D Pointcheval, J Stern. Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology, 2000,13(3): 361-396. (上文的长文版本)

M. Bellare, C. Namprempre. G. Neven. Security Proofs for Identity-based Identification and Signature Schemes, Eurocrypto04, LNCS3027, 2004: 468-486 (基于身份的签名的模块化证明方法)

J. Coron. On the Exact Security of Full Domain Hash. Crypto00, LNCS1880, Springer, 2000: 229-235 (紧规约,精确安全性)

J. Coron. Optimal Security Proofs for PSS and other Signatgure Schemes. Eurocrypto02, LNCS 2332, Springer, 2002: 272-287 (紧规约)

E. Goh, S. Jarechi. A Signature Scheme as Secrure as the Diffie-Hellman Problems. Eurocrypt03, LNCS2656, Springer, 2003: 401-415 (紧规约)

J. Katz, N. Wang. Efficiency Improvements for Signature Schemes with Tight Security Reductions. CCS03, 2003: 155-164 (紧规约)

R. Gennaro, S. Halevi, T. Rabin. Securei Hash-and-Sigh Signtaures with the Random Oracle. Eurocrypt99, LNCS1592, Springer, 1999: 123-139 (第一个具有实用价值的标准模型下的可证明安全签名,基于强 RSA 假设)

R. Cramer, V. Shoup. Signature Schemes based on the Strong RSA Assumptin. ACM TISSEC, 2000,3(3): 161-185 (更为实用的标准模型的安全签名)

D. Boneh, X. Boyen. Short Signatures with Random Oracles. Eurocrypt04, LNCS3027, Springre, 2004: 382-400(第一个基于双线性对,强 CDH 假设,标准模型下的安全签名)

K. G. Paterson, JCN Schuldt. Efficient Identity-based Signatures Secure in the Standard Model. ACISP06, LNCS4058, Springer, 2006: 207-222.

协议

D. Dolev, A. C. Yao. On the Security of Public Key Protocols. FOCS81, 1981: 350-357 (Dolev-Yao 模型,最早的形式化协议分析)

M. Bellare, P. Gogaway. Entity Authentication and Key Distribution. CRYPTO'93, Berlin, Springer-Verlag, 1993: 232-249.(提出基于复杂性理论的通信模型,给出了安全认证的密钥建立的形式化定义)

R. Canetti. Universally Composable Security: A New Paradigm for Cryptography Protocols. FOCS01, 2001: 136-145(UC 安全)

U. Feige, A. Shamir. Witness Indistinguishable and Witness Hiding Protocols. STOC 90, 1990: 416-426 (提出证据不可区分和证据隐藏的概念)

参 考 文 献

[1] 陈鲁生,等.现代密码学.北京:科学出版社,2002.
[2] 陈恭亮.信息安全数学基础.北京:清华大学出版社,2004.
[3] 陈少真.密码学基础.北京:科学出版社,2008.
[4] 曹珍富.公钥密码学.哈尔滨:黑龙江教育出版社,1993.
[5] 曹天杰,等.安全协议.北京:北京邮电大学出版社,2009.
[6] 邓安文.密码学——加密演算法.北京:中国水利水电出版社,2006.
[7] 丁存生,肖国镇.流密码学及其应用.北京:国防工业出版社,1994.
[8] 裴定一,等.算法数论.北京:科学出版社,2002.
[9] 范九伦,等.密码学基础.西安:西安电子科技大学出版社,2008.
[10] 冯登国,裴定一.密码学导引.北京:科学出版社,1999.
[11] 冯登国.密码分析学.北京:清华大学出版社,南宁:广西科学技术出版社,2000.
[12] 冯登国,吴文玲.分组密码的设计与分析.北京:清华大学出版社,2000.
[13] 冯登国.网络安全原理技术.北京:科学出版社,2003.
[14] 冯晖,等.计算机密码学.北京:中国铁道出版社,1999.
[15] 谷利泽,等.现代密码学教程.北京:北京邮电大学出版社,2009.
[16] 胡向东,等.应用密码学教程.北京:电子工业出版社,2005.
[17] 何大可,等.现代密码学.北京:人民邮电出版社,2009.
[18] 李克洪,王大玲,董晓梅.实用密码学与计算机数据安全.沈阳:东北大学出版社,1997.
[19] 李顺东,等.现代密码学:理论、方法与研究前沿.北京:科学出版社,2009.
[20] 刘嘉勇,等.应用密码学.北京:清华大学出版社,2008.
[21] 刘木兰.密钥共享体制和安全多方计算.北京:电子工业出版社,2008.
[22] 林东岱,等.应用密码学.北京:科学出版社,2009.
[23] 林东岱.代数学基础与有限域.北京:高等教育出版社,2006.
[24] 卢开澄.计算机密码学——计算机网络中的数据保密与安全.2版.北京:清华大学出版社,1999.
[25] 毛明,等.大众密码学.北京:高等教育出版社,2005.
[26] 潘承洞,等.初等数论.北京:北京大学出版社,2003.
[27] 邱卫东,等.密码协议基础.北京:高等教育出版社,2009.
[28] 卿斯汉.安全协议.北京:清华大学出版社,2005.
[29] J. Katz, Y. Lindel. Introtroduction to Modern Cryptography—Principle and Protocok,现代密码学——原理与协议.任伟,译.北京:国防工业出版社,2010.
[30] 任伟.现代密码学.2版.北京:北京邮电大学出版社,2014.
[31] 孙淑玲.应用密码学.北京:清华大学出版社,2004.
[32] 宋震,等.密码学.北京:中国水利水电出版社,2002.
[33] 沈世镒.组合密码学.杭州:浙江科学技术出版社,1992.
[34] 沈世镒.近代密码学.桂林:广西师范大学出版社,1998.
[35] 田园.计算机密码学——通用方案构造及安全性证明.北京:电子工业出版社,2008.
[36] 王亚弟,等.密码协议形式化分析.北京:机械工业出版社,2006.
[37] 王育民,何大可.保密学——基础与应用.西安:西安电子科技大学出版社,1990.

[38] 王育民,梁传甲.信息与编码理论.西安:西北电讯工程学院出版社,1986.
[39] 王育民,刘建伟.通信网的安全——理论与技术.西安:西安电子科技大学出版社,1999.
[40] 王新梅,马文平,武传坤.纠错密码理论.北京:人民邮电出版社,2001.
[41] 王亚弟,等.密码协议形式化分析.北京:机械工业出版社,2007.
[42] 万哲先.代数和编码.3版.北京:高等教育出版社,2007.
[43] 徐茂智.信息安全基础.北京:高等教育出版社,2006.
[44] 徐茂智,游林.信息安全与密码学.北京:清华大学出版社,2007.
[45] 肖攸安.椭圆曲线密码体系研究.武汉:华中科技大学出版社,2006.
[46] Hans Delfs,Helmut Knebl.密码学导引:原理与应用.肖国镇,张宁,译.北京:清华大学出版社,2008.
[47] 杨波.现代密码学.北京:清华大学出版社,2003.
[48] 杨义先,等.现代密码新理论.北京:科学出版社,2002.
[49] 杨义先,等.应用密码学.北京:北京邮电大学出版社,2005.
[50] 杨义先,等.网络安全理论与技术.北京:人民邮电出版社,2003.
[51] 杨晓元.现代密码学.西安:西安电子科技大学出版社,2009.
[52] 赵泽茂.数字签名理论.北京:科学出版社,2007.
[53] 张福泰,等.密码学教程.武汉:武汉大学出版社,2006.
[54] Hankerson D,Menezes A,Van stone S.椭圆曲线密码学导论.张焕国,等译.北京:电子工业出版社,2003.
[55] 张焕国,刘玉珍.密码学引论.武汉:武汉大学出版社,2004.
[56] 张世永.网络安全原理与应用.北京:科学出版社,2003.
[57] 章照止.现代密码学基础.北京:北京邮电大学出版社,2004.
[58] 钟义信.信息科学原理.北京:北京邮电大学出版社,1996.
[59] 周荫清.信息理论基础.北京:北京航空航天大学出版社,1993.
[60] 周炯槃.信息理论基础.北京:人民邮电出版社,1984.
[61] 郑东,等. 密码学——密码算法与协议.北京:电子工业出版社,2009.
[62] 祝跃飞,等.公钥密码学设计原理与可证明安全.北京:高等教育出版社,2010.
[63] 祝跃飞,等.密码学与通信安全基础.武汉:华中科技大学出版社,2008.
[64] A. Menezes,等.Handbook of Applied Cryptography.胡磊,等译.应用密码学手册.北京:电子工业出版社,2005.
[65] Bruce Schneier.应用密码学——协议、算法与C源程序.吴世忠,等译.北京:机械工业出版社,2000.
[66] Douglas R. Stinson.密码学原理与实践.冯登国,译.北京:电子工业出版社,2003.
[67] Ranjan Bose.信息论、编码与密码学.武传坤,译.北京:机械工业出版社,2005.
[68] Niels Ferguson,等.密码学实践.张振峰,等译.北京:电子工业出版社,2005.
[69] Paul Garrett.密码学导引.吴世忠,宋晓龙,郭涛,等译.北京:机械工业出版社,2003.
[70] Richard Spillman.经典密码学与现代密码学.叶阮健,等译.北京:清华大学出版社,2005.
[71] Ross J. Anderson.信息安全工程.蒋佳,刘新喜,等译.北京:机械工业出版社,2003.
[72] Thomas M. Cover.信息论基础(影印本).北京:清华大学出版社,2003.
[73] Trappe W, Wahington L C. 密码学与编码理论.王金龙,等译.北京:人民邮电出版社,2008.
[74] Wenbo Mao.现代密码学理论与实践.王继林,等译.北京:电子工业出版社,2004.
[75] William Stallings.密码编码学与网络安全——原理与实践.4版.孟庆树,王丽娜,傅建明,等译.

北京：电子工业出版社，2007.
[76] Wade Trappe，Lawrence C Washington. 密码学概论. 邹红霞，等译. 北京：人民邮电出版社，2004.
[77] Neal Koblitz. A Course in Number Theory and Cryptography. 2nd ed. Springer，GTM，1994.
[78] Oded Goldreich. Foundation of Cryptography，Basic Tools. 北京：电子工业出版社，2003.
[79] Oded Goldreich. Foundation of Cryptography，Basic Applications. 北京：电子工业出版社，2005.
[80] 中国密码学会. 密码学学科发展报告 2009—2010. 北京：中国科学技术出版社，2010.
[81] 中国密码学会. 中国密码学发展报告 2008. 北京：电子工业出版社，2009.
[82] 中国密码学会. 中国密码学发展报告 2009. 北京：电子工业出版社，2010.
[83] 中国密码学会. 密码术语. 北京：电子工业出版社，2009.